Hindawi Publishing Corporation
International Journal of Computer Games Technology
Volume 2016, Article ID 8492312, 9 pages
http://dx.doi.org/10.1155/2016/8492312

Research Article

Visualizing Changes in Strategy Use across Attempts via State Diagrams: A Case Study

Deirdre Kerr

Center for Advanced Psychometrics, Educational Testing Service, San Francisco, CA 94105, USA

Correspondence should be addressed to Deirdre Kerr; dkerr@ets.org

Received 28 September 2015; Revised 6 January 2016; Accepted 12 January 2016

Academic Editor: Francisco Jose Seron

Game log data have great potential to provide actionable information about the in-game behavior of players. However, these low-level behavioral data are notoriously difficult to analyze due to the challenges associated with extracting meaning from sparse data stored at such a small grain size. This paper describes a three-step solution that uses cluster analysis to determine which strategies players use to solve levels in the game, sequence mining to identify changes in strategy across multiple attempts at the same level, and state transition diagrams to visualize the strategy sequences identified by the sequence mining. In the educational video game used in this case study, cluster analysis successfully identified 15 different in-game strategies. The sequence mining found an average of 40 different sequences of strategy use per level, which the state transition diagrams successfully displayed in an interpretable way.

1. Introduction

Game log data provide an opportunity to collect data about players' problem-solving strategies and mistakes [1–3] and can do so in an unobtrusive manner [4, 5] that is both feasible and cost-effective [6].

Because log data from games record the exact behavior of players [7], they can provide useful information about players' thought processes, intentions, and abilities. For example, log data can be used to provide detailed measures of the extent to which players have mastered specific goals [8] or to support diagnostic claims about players' processes [9]. These data also have the potential to be used to identify the strengths and weaknesses of individual players [10], provide individualized feedback [11], allow players to track their own progress [3], guide instruction that is optimal for each player [12], or modify presentation order for different player groups [13, 14].

Though log data are very comprehensive and detailed, the inclusion of such fine-grained detail presents a number of problems for analysis [5, 15]. First, log files contain large quantities of information, typically consisting of thousands of pieces of information on each player [16], with a single player able to generate over 3,000 actions in just half an hour of game play [17]. Second, the data are at such a small grain size (e.g., "selected tool A," "moved game character to the left") that there is often no known theory to help identify precisely which actions are important to the construct being measured [18]. Third, examining fine-grained behaviors in such highly unstructured environments creates some inherent problems [19]. This is because log data are sparse (in that any given player produces a lot of actions, but any given action may only be produced by a few players), noisy (in that irrelevant actions can vastly outnumber relevant ones, and relevant actions are not usually identifiable a priori), and so large that they are prohibitively costly to examine by hand.

Because the log files generated by video games or simulations are too large to analyze manually [15], data mining techniques must be used to identify and describe meaningful patterns despite the noise surrounding them [20, 21]. Extracting implicit, interesting patterns from large data sets can identify unexpected behavioral patterns that can lead to the discovery of new knowledge about how players solve problems [22] and can allow questions to be addressed which were not previously feasible to answer [23].

In this paper, data mining and statistical techniques are combined to produce visualizations that can be used to discover interesting patterns of in-game behavior that have implications for in-game learning and game design. This

three-layer process first uses cluster analysis to identify the different strategies players use to solve levels in the game. Then, sequence mining is used to identify frequent changes in strategy use. Finally, state transition diagrams are used to visualize the sequences so that they are more easily interpreted.

The methods are presented in Section 2, including a description of the game used in the case study and the data structure and preprocessing that supported the analyses. In Section 3, the results of the cluster analysis, sequence mining, and state transition diagrams are presented, and the importance of these findings for game design and player modeling is discussed in Section 4.

Figure 1: Screenshot of *Save Patch*.

2. Methods

This study chains three different methods together to produce visualizations that can be used to discover interesting patterns of in-game behavior that have implications for in-game learning and game design. To guide the discussion of these methods, a detailed description of the educational video game used in the case study is first provided. While the use of an educational video game as the case study is not intended to limit application of these methods solely to educational games, it is important to note that the methods used in this study may have limited use in games with fundamentally different organization. Notably, these methods assume a game with multiple levels wherein multiple attempts are allowed for each level and the levels are nested in stages of similar levels (i.e., levels sharing the same mechanics, affordances, and goals).

After discussion of the game description in Section 2.1, Section 2.2 describes the structure of the log data from the game and the preprocessing steps necessary to support analysis. The three methods are then described in Section 2.3 (cluster analysis), Section 2.4 (sequence mining), and Section 2.5 (state transition diagrams).

2.1. Game Description. The game used in this study is *Save Patch*, an educational video game designed by the National Center for Research on Evaluation, Standards, and Student Testing (CRESST) at the University of California, Los Angeles, and the Game Innovation Lab at the University of Southern California.

Save Patch, pictured in Figure 1, is a puzzle game about fractions. In *Save Patch*, players must correctly identify the fractional units represented in the level, break up the whole unit ropes in the Path Options bin into the appropriate fractional size, and drag the necessary number of fractional pieces of rope to each sign on the game grid in order to guide their character safely from sign to sign until the prize is reached. When the player has finished adding the desired number of rope pieces to each sign on the grid, the player presses the Go button to determine whether the game character will successfully reach the prize.

The game area is represented as a single linear dirt path in one-dimensional levels and a dirt path grid in two-dimensional levels. Places where a dirt path intersects another dirt path represent unit markers, and small red posts along the dirt path indicate the fractional pieces each unit is broken into. The level pictured in Figure 1 is a one-dimensional level, that is, two units across. Each unit is broken into thirds. The distance between the first and second sign is 3/3, the distance between the second and third sign is 1/3, and the distance between the third sign and the prize is also 1/3.

This case study examined log data generated by 855 students from 31 classes who attempted to complete all 23 levels of *Save Patch* in approximately one hour of game time spread across two separate class periods. The resulting data set consisted of 1,288,103 individual entries.

2.2. Data Structure and Preprocessing. In order to record in-game actions that might be indicative of players understanding of fractions, the data generated by the game were logged in the format described in detail in [24]. While video games are often designed to log every mouse click, we chose to ignore mouse clicks or drags that did not result in an action in the game (e.g., mouse clicks on the game background) and to capture only mouse clicks that represented game state changes or deliberate student actions such as clicking on a rope, dragging a rope to a sign on the grid, or clicking on the "Reset" button.

Additional structure was added by assigning codes to each of the different types of actions that could occur in the game, such as selecting a rope (code 3000) or adding a rope to a sign (code 3010). Codes 1000–1999 were used for general game information, such as game version or study condition; 2000–2999 were used for basic manipulation of objects, such as toggling a fraction to a new denominator; 3000–3999 were used for in-game mathematical decisions, such as adding a fraction to a sign; 4000–4999 were used for success or failure states such as player deaths or event-driven feedback; 5000–5999 were used for in-game navigation such as returning to the main menu or advancing to the next level; and 6000–6999 were used for the help menu system.

The uniqueness of events was preserved by including columns in the log data capturing the specific detail of each event, along with a description of how to interpret the data. Thus, each action was captured at both a general and a specific level, as can be seen by the logging in Table 1 of a set of student actions resulting in the addition of 2/3 to the leftmost sign on the grid.

In order for the log data to be used in cluster analysis or other data mining techniques, the log data in Table 1 first had

Table 1: Example *Save Patch* log data.

Player ID	Game time	Data code	Data description	Data 01	Data 02	Data 03
1115	3044.927	2050	Scrolled rope from [value] to [value]	1/1	3/3	
1115	3051.117	3000	Selected rope of value [value]	1/3		
1115	3054.667	3010	Added [value] to [position] resulting in [value]	1/3	1/0	1/3
1115	3058.443	3000	Selected rope of value [value]	1/3		
1115	3064.924	3010	Added [value] to [position] resulting in [value]	1/3	1/0	2/3

to be transformed into a structure more amenable to those techniques. The first step in this process was to calculate a mnemonic summarizing all relevant information in each row. The mnemonic consisted of two parts. The first part was a word indicating the type of action being taken (e.g., SCROLL, SELECT, and ADD) and the second part was one or more values with short indicators of the context of each additional value beyond the first (e.g., GET_1o3).

All blank spaces in the mnemonics were removed and all symbols were replaced with letters to allow the mnemonic to function as a variable name in a variety of statistical software. For example, the symbol "/" was replaced with "o" (for "over") and "-" was replaced with "n" (for "negative"). The mnemonics for the data in Table 1 were SCROLL_1o1_TO_3o3, SELECT_1o3, ADD_1o3_TO_1o0_GET_1o3, SELECT_1o3, and ADD_1o3_TO_1o0_GET_2o3.

The data were then transformed into a sparse binary matrix wherein each row represented a single student's attempt to solve the level (where an attempt was defined as the set of actions beginning immediately after either a new level load or a level reset and ending at either a level reset or a level completion) and each column represented a possible action that could be taken in that level. 1's indicated actions made in a given attempt and 0's indicated actions not made in that attempt. The resulting matrix was very sparse, with only ten to twenty of the hundreds of possible actions being performed by a given player in a given attempt.

2.3. Cluster Analysis. Cluster analysis is a density estimation technique for identifying patterns reflecting differences in attitudes, thought processes, or behaviors [22, 25] through the analysis of either general or sequential correlations [20].

Cluster analysis partitions entities into groups on the basis of a matrix of interobject similarities [26] by minimizing within-group distances compared to between-group distances so that entities classified as being in the same group are more similar to each other than they are to actions in other groups [27]. Using these similarities, cluster analysis algorithms can identify the latent grouping structure of the data [28, 29] and perform the necessary pattern reduction and simplification, so the patterns present in large data sets can be detected [30].

Groups of commonly cooccurring in-game actions in *Save Patch* were identified using the fuzzy feature cluster analysis algorithm *fanny* [31] in *R* [32]. The most common distance calculation used in cluster analysis is the Euclidean or Squared Euclidean distance because data are often either continuous or ordinal. However, in the analysis of *Save Patch*, the data were binary, indicating whether or not a given player performed a given action in a given attempt, so the Manhattan distance was used instead [33].

In a standard cluster analysis approach, the result is the identification of distinct groups of players that differ in how they play the game. However, in such analyses, the features of game play that distinguish the player groups would not be known. For that reason, feature cluster analysis was run instead.

Feature clustering differs from standard clustering techniques in that rather than clustering entities in feature space it clusters features in entity space. Feature clustering does not require different algorithms from standard clustering and can be run by simply transposing the matrix being operated on [34]. Therefore, in this analysis, actions were considered to be similar if they were frequently performed by the same players in the same attempt and different if they were never performed by the same player in the same attempt (rather than considering players to be similar if they performed the same actions and different if they performed different actions).

The analysis of *Save Patch* also differed from most cluster analyses used in educational data mining in that it used fuzzy cluster analysis [35] rather than hard cluster analysis. Hard clustering forces each action to belong to a single cluster, whether or not all actions are easily classified in this manner. Game log files are likely to require the use of fuzzy clustering algorithms because different in-game strategies often contain some of the same initial actions. Fuzzy cluster analysis would allow each such action to belong to both clusters, using probability theory to identify the proportion of belongingness in each cluster. This allows for superior clustering results for data with problematic data points lying between otherwise easily identifiable clusters [35]. While fuzzy clustering and hard clustering will return similar results if the data are not very fuzzy, the fuzzier the data are, the more imprecise the hard clustering results will be [36].

Clustering algorithms provide no single definitive method of determining the number of clusters present in the data. Rather, solutions with different numbers of clusters must be compared using both statistical and substantive criteria. Therefore, the application of cluster analysis in this study required human judgment to determine when the correct number of clusters was reached.

In order to determine the correct number of clusters in each level in the fuzzy cluster analysis, each level was run with two clusters, then three clusters, then four, and so on until one of two things occurred: incorrect actions began to appear in the cluster representing the anticipated solution for that level, or the additional cluster provided no additional

interpretive value (e.g., the additional cluster resulted in the split of an easily identifiable strategy into two parts). Once either of those outcomes occurred, cluster analysis for that level ceased and the largest number of previously successful clusters was retained. For example, if incorrect actions began to appear in the cluster representing the anticipated solution to the level when a level was run with eight clusters, it was determined that seven was the optimal number of clusters for that level.

These decisions made the results of the cluster analysis more interpretable, in that each cluster consisted of a list of actions that were frequently performed together and common preliminary actions (such as scrolling) were allowed to belong to multiple clusters. Due to these decisions, it was trivial to look at each resulting cluster and identify through human judgment both the strategy being used and the key actions making up that strategy. For example, the cluster representing the anticipated solution to the level pictured in Figure 1 would have to include ADD_1o3_TO_0o0_GET_3o3, ADD_1o3_TO_0o3_GET_1o3, and ADD_1o3_TO_0o4_GET_1o3 because the correct solution to the level is 3/3 on the first sign, 1/3 on the second sign, and 1/3 on the third sign. While the identified cluster would consist of other actions such as ADD_1o3_TO_0o0_GET_2o3, only a subset of the listed actions would be necessary to uniquely identify the strategy being performed (as it would be impossible to add 1/3 to a sign and get 3/3 if one had not previously added 1/3 to get 2/3).

Therefore, the final step in determining the strategy used in each attempt was to write a series of *if* statements reflecting the uniquely identifying actions for each cluster and then use those statements to code each attempt with the strategy represented therein. Attempts that did not match the identified action sets were coded as *unknown error*, assuring that all attempts (even ones we could not explain) were assigned a strategy.

For a more detailed description of this process and its benefits over standard applications of cluster analysis for analyzing game data, see [37].

2.4. Sequence Mining. Cluster analysis ignores certain salient aspects of the log data, such as timing and order [38]. For example, if a player performed Action A, then Action B, and then Action C and another player performed Action C, then Action B, and then Action A, both players would be in the same cluster because they performed all the same actions. If timing or order is important to the analysis, then sequence mining must be used instead of or in addition to cluster analysis. Sequence mining is a technique for identifying patterns within user actions that frequently occur in the same order [39].

Sequence mining can identify previously unknown misconceptions [40] or patterns of behavior that can be used to support or improve decision making [41]. However, it produces a prohibitively large number of sequences [40] that are often redundant or difficult to understand [15]. Therefore, even though order is very important in game data (particularly data from educational games) because it often indicates increasing difficulty [42], time-based analyses of such data are rare [43].

In order to determine the change in strategy use over time, the results of the cluster analysis were transformed into the format required by sequence mining (e.g., one row per player per level and one column per attempt). Each cell in the resulting table consisted of the strategy that player used in that attempt to solve the level.

Frequent patterns of strategy use were identified from this data set using the sequence mining algorithm cspade [44] in *R* [32]. In order to remove rare events from the analysis, strategies that occurred in less than 2% of attempts were removed from the analysis.

To increase interpretability, the minimum and maximum time difference between consecutive baskets were both set to 1.0 so that consecutive strategies in a strategy sequence would indicate strategies used in consecutive attempts rather than strategies used in some later attempt. This was a particularly important decision because in the majority of cases each student eventually reached a solution to the level (and, therefore, all strategies eventually lead to a solution), but only direct transitions were of interest in this analysis.

The output of the sequence mining was a list of the identified sequences ordered first by sequence length (sequences of length two, followed by sequences of length three, and so on) and then by the frequency of occurrence.

2.5. State Transition Diagrams. Sequence mining is notorious for producing an overwhelming number of sequences, which can be prohibitively difficult to interpret in their standard list format [15, 40]. This is partially because the standard list format makes it difficult to examine all the sequences related to a specific strategy at once, since the sequences cannot easily be sorted by strategy because the strategy of interest can occur at any point in the sequence. However, if each strategy is thought of as being a state the player is temporarily in, state transition diagrams can be used to visualize the sequences in a way that supports interpretation of the relationship between a given strategy and subsequent behavior.

State transition diagrams are directed graphs that represent all possible states of a given system. Each node in the diagram represents a state, and arrows between nodes indicate possible state transitions [45].

In most state transition diagrams, each transition is labeled with the triggering event that would move the system from one state to the other. However, if the states are player strategies, a player can move from one state to another on each new attempt and could theoretically move from any possible state to any other possible state (since the triggering event is the same for all states).

State transition diagrams already suffer from interpretability problems when the number of states is large and there are multiple states that could be reached from any given state. A fully interconnect state transition diagram would be almost entirely uninterpretable.

Therefore, in this study, the results of the sequence mining are used to indicate which states are likely to transition to which other states. State transitions that are too infrequent to be captured by the sequence mining (e.g., less than 10% of

sequences that contain the triggering event transition to the given state) are not shown in the diagram. One diagram was produced per level, containing only the frequent states (rather than all possible states) and only the frequent transitions (rather than all possible transitions). Rather than labeling each transition with the triggering event (i.e., the initial state), additional information was added to the state transition diagram by including the percentage of players in the initial state who transitioned to the second state in their next attempt at the level.

Additionally, extended state transition diagrams were created for sequences longer than two. In these diagrams, the node for the initial state indicated the sequence of states leading up to the last state transition represented in the sequence (e.g., for the sequence "A," "B," and "C," the initial nodes would be "A to B"). The transition label would then indicate the percentage of players who had performed the "A to B" sequence that then performed the "C" strategy.

3. Results

The cluster analysis results in this study come from an earlier analysis of the same *Save Patch* data [37], wherein 15 different strategies for solving levels were identified. Those results are reported first so that the description of the sequence mining results and state transition diagrams would be understandable. After these earlier cluster analysis results are reported, the results from the sequence mining and state transition diagrams run in this study are reported.

Sequence mining of the changes in strategy use over time identified 916 different sequences of strategy use, indicating that some players continued to make the same errors, some switched from one error to another, and some moved from an error to a valid solution strategy. State transition diagrams were then used to visualize the strategy sequences to support interpretation of the relationship between a given strategy of interest and subsequent in-game behavior.

3.1. Cluster Analysis. The cluster analysis of this data led to the identification of six different types of valid solution strategies and nine different types of errors in *Save Patch*. The *standard solution* was the anticipated solution for each level. A *fractional solution* differed from the standard solution only in that whole units were represented as their fractional equivalents. An *alternate solution* was a valid solution using a denominator other than the one represented in the level. A *shortcut solution* was a valid solution that skipped one or more signs. An *incomplete solution* occurred when a player placed ropes correctly on one or more signs but left one or more signs empty. A *reset solution* occurred when a player placed ropes correctly on all signs but then hit reset rather than submitting their answer.

Most of the nine strategies that resulted in errors were mathematical misconceptions involving the three main skills required for adding fractions: unitizing, partitioning, and iterating. Unitizing consists of correctly identifying the number of units represented, partitioning consists of correctly identifying the number of pieces each unit is broken into (e.g., the denominator of the represented fraction), and iterating consists of identifying the number of unit fractions in the representation (e.g., the numerator of the represented fraction) [46].

Players who made *unitizing errors* were unable to determine the number of units being represented. These players either saw the entire representation as a single unit, regardless of the number of units represented, or saw each fractional piece as a whole unit.

Players who made *partitioning errors* were unable to determine the number of pieces the unit was broken into. These students either counted dividing marks to determine the number of pieces the unit was broken into (e.g., seeing halves instead of thirds) or counted dividing marks and unit marks (e.g., seeing fourths instead of thirds). Some players even combined both errors, *unitizing and partitioning*, seeing the entire representation as one unit and counting dividing marks to determine the number of pieces the unit was broken into.

Players who made *iterating errors* were unable to determine the numerator of the represented fractions. These players were able to correctly identify the number of units being represented and the number of pieces each unit was broken into but were unable to determine the number of pieces to put on each sign.

Additionally, the cluster analysis identified two game-related errors in which players used the correct mathematical strategies but moved in the *wrong direction* or *skipped signs* that were mandatory (indicated by the color of the sign, e.g., the second sign in Figure 1). Other players avoided mathematics entirely by placing all the available resources on the signs in the order that they were given in the resource bin (the *everything in order* strategy). All attempts that did not fall into one of the preceding strategies were identified as *unknown error* (generally 5% to 15% of attempts in a given level).

3.2. Sequence Mining. Sequence mining of the strategies identified by the cluster analysis found an average of 40 sequences of length 3 or less per game level, with the number of sequences identified in a given level ranging from 7 to 68. In total, there were 916 sequences identified, with each student performing a maximum of 23 of those sequences (one for each level of the game).

Given that there were 855 players and 916 sequences, analysis of individual sequences was not feasible. Therefore, in order to reduce the sequences to a manageable number, sequences of length two were reported in the state transition diagrams, but sequences longer than length two that demonstrated similar behavior were recoded into a single category before being reported in the extended state transition diagrams (e.g., players who made two partitioning errors in a row were combined with players who made three partitioning errors in a row into a single category called *partitioning to partitioning*).

This allowed for examination of the effect of specific strategy sequences. For example, in this study, we decided to look at the impact of making partitioning errors. To do so, players were categorized into one of six groups according to their partitioning behavior on the 11 levels in the game that targeted partitioning errors.

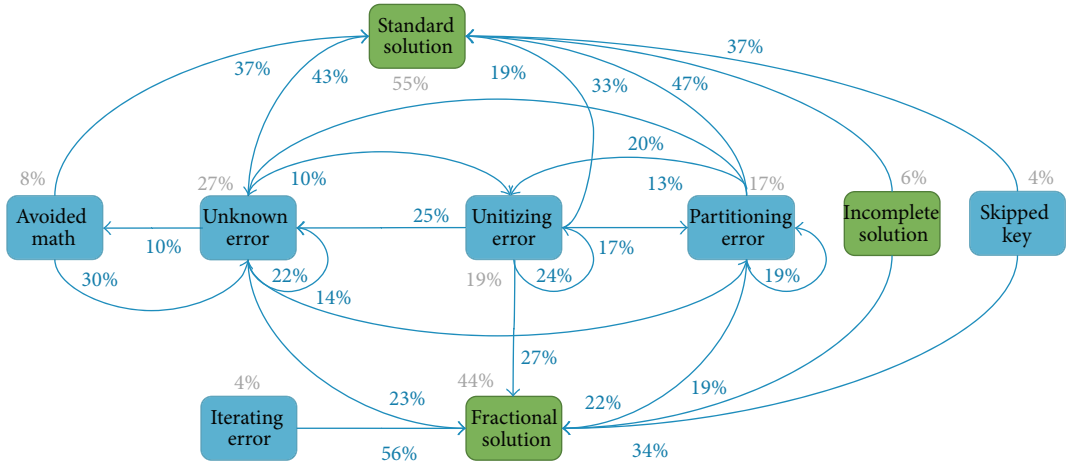

Figure 2: State transition diagram for Level 9.

There were two different categories that did not involve partitioning. Players either used only valid solution strategies (e.g., made no partitioning errors) in all attempts at all levels of the game addressing partitioning (the *all correct* sequence, consisting of 3% of players) or moved from errors that did not involve partitioning to valid solution strategies (*no partitioning errors*, 9% of players).

There were three different categories of sequences reflecting movement into or out of partitioning errors. Players either repeatedly made partitioning errors (*repeated partitioning*, 29% of students), moved from making partitioning errors to errors not involving partitioning (*abandoned partitioning*, 27% of players), or moved from partitioning errors to valid solution strategies (*corrected partitioning*, 23% of players).

Overall, using sequence mining to identify in-game strategy sequences resulted in substantively meaningful categorizations of 91% of behavior in the 11 game levels addressing partitioning.

3.3. State Transition Diagrams. Graphing the entire set of sequences identified in a given level in a state diagram allowed for the identification of patterns of behavior that do not need to be identified by hand from the sequence mining results, as done in the previous section with the coding of the different types of partitioning sequences. As an example, the state transition diagram for Level 9 is shown in Figure 2.

In this figure, blue nodes represent erroneous strategies and green nodes represent valid solution strategies. The standard solution and fractional solution nodes are terminal, in that either of those strategies will advance the player to the next level. The percentage of total attempts corresponding to use of each strategy in the level is indicated in the gray number adjacent to each node. The percentage of players moving from one strategy to another is indicated in the blue number adjacent to each arrow between nodes (e.g., unitizing errors accounted for 19% of attempts in the level, and 17% of the occurrences of a unitizing error were followed immediately by a partitioning error).

Note that it is possible for the percentages for a given node to total to more than 100%. This is because it is possible (and, in fact, quite common) for a given player to use a given strategy more than once. Therefore, a player performing two unitizing errors followed by a partitioning error would count in both the 24% of players moving from a unitizing error to another unitizing error and the 17% of players moving from a unitizing error to a partitioning error.

This figure allows for the identification of some interesting patterns of in-game behavior. For example, the players who avoided mathematics either got the standard solution on their next attempt or made an unknown error. It is notable that there is no strategy sequence that connects the avoided mathematics error to any of the other errors. This indicates that there is a subset of players who are not playing the game as a mathematics game, as they are not using mathematical strategies to solve the levels. These players are clearly not behaving as anticipated, and modifications to the game to reduce this behavior should be considered.

These state transition diagrams work well for sequences of length two but quickly become too crowded with longer sequences. Therefore, these sequences were examined in the extended state diagrams.

Figure 3 shows the extended state diagram for the level pictured in Figure 2. Only sequences of length three are displayed because no sequences of length four were found for this level. The purple nodes indicate the first two strategies in the strategy sequences of length three. The blue nodes indicate the last strategy in the strategy sequences of length three. The purple arrows link the first two strategies in a three strategy sequence with the last strategy in that sequence. The purple numbers above those arrows indicate the percentage of students who performed the first two strategies in the sequence that then performed the third strategy in the sequence. The blue dashed arrows are carried up from the original state transition diagram to indicate the percentage of students who performed a given strategy that then performed the given two-strategy sequence.

This figure indicates that players are more prone to repeating (more than twice in a row) the unitizing error or the unknown error than the partitioning error. This is indicated by the fact that a quarter of the students who made a unitizing

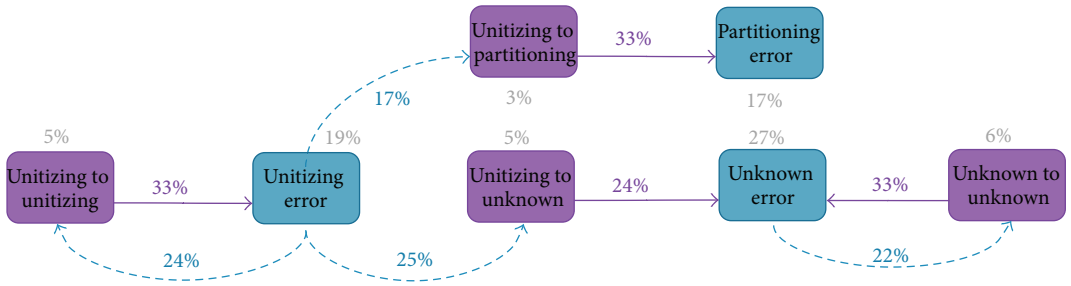

Figure 3: Extended state transition diagram for Level 9.

error immediately made another unitizing error, and a third of the students who made two unitizing errors in a row went on to make a third unitizing error. A similar pattern can be seen with the unknown error. However, no such pattern can be seen with the partitioning error (where the *partitioning to partitioning* sequence does not show up in any sequence of length three and is therefore not included in the extended state transition diagram). This indicates that players might need help in figuring out what they are doing wrong if they are making unitizing errors or unknown errors but that no such assistance is required for players making partitioning errors, as the behavior is self-correcting.

Note that while the state transition diagrams and extended state transition diagrams were generated by hand for this study, it would be fairly easy to import the sequence mining results as triples into existing programs for graphing triples such as [47]. While the relatively small number of sequences identified in the data set used in this case study did not require automated generation of the state transition diagrams, it is something that would clearly be necessary for larger data sets.

4. Conclusion

The identification of in-game strategies that allows for the development of a deeper understanding of how players solve in-game problems, as previous researchers posited, would be possible once in-game behavior could be measured [1–3].

First, the identification of a variety of correct solution strategies provides detail about how successful players reached their answers, rather than just identifying such players as correctly or incorrectly solving the game levels.

Second, the identification of specific strategies reflecting why and how players fail to solve game levels could be used to provide targeted remediation (e.g., providing information about how to identify the denominator to players who make partitioning errors or providing reminders about how to change the direction of the signs to players who repeatedly moved in the wrong direction) or support diagnostic claims about player understanding (e.g., not only identifying that a given player does not know the correct answer, but also determining which specific skills they lack).

Finally, the identification of strategies reflecting game errors allows for the separation of the impact of the game design from player understanding of the content area. For example, players using the wrong direction strategy were mathematically correct but did not solve the level on that attempt because they made a game-related error.

Examining in-game strategy sequences could allow game designers to identify which players are struggling and which players are gaining mastery. This could allow them to provide individualized in-game feedback for players who continue to struggle with a specific concept or game mechanic [11] or create game levels that adapt to the changing needs and abilities of each player [12–14]. Additionally, examining the differences between strategy sequences in different portions of the game could indicate problematic levels or game areas where certain undesirable sequences are more likely.

Producing state transition diagrams not only allows for easier examination of the identified sequences but also provides necessary information for use in modeling techniques such as Markov chains. This is because the values in the state transition diagrams identify and establish the relative frequency of the state transition probabilities required by the model [47].

In sum, the results of this study support the use of game log data to extract actionable information about the in-game behavior of players by extracting and visualizing features indicative of in-game strategy use and changes in strategy use over time.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This material is based upon work supported by the Institute of Education Sciences, US Department of Education, through Grant R305C080015 to the National Center for Research on Evaluation, Standards, and Student Testing (CRESST).

References

[1] A. Merceron and K. Yacef, "Mining student data captured from a web-based tutoring tool: initial exploration and results," *Journal of Interactive Learning Research*, vol. 15, no. 4, pp. 319–346, 2004.

[2] E. S. Quellmalz and J. W. Pellegrino, "Technology and testing," *Science*, vol. 323, no. 5910, pp. 75–79, 2009.

[3] M. Rahkila and M. Karjalainen, "Evaluation of learning in computer based education using log systems," in *Proceedings of the 29th ASEE/IEEE Frontiers in Education Conference (FIE '99)*, pp. 16–22, San Juan, Puerto Rico, November 1999.

[4] J. H. Kim, D. V. Gunn, E. Schuh, B. Phillips, R. J. Pagulayan, and D. Wixon, "Tracking real-time user experience (TRUE): a comprehensive instrumentation solution for complex systems," in *Proceedings of the 26th Annual SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*, pp. 443–452, ACM, Florence, Italy, April 2008.

[5] J. Mostow, J. Beck, A. Cuneo, E. Gouvea, C. Heiner, and O. Juarez, "Lessons from project LISTEN's session browser," in *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. S. J. D. Baker, Eds., pp. 389–416, CRC Press, Boca Raton, Fla, USA, 2011.

[6] E. S. Quellmalz and G. Haertel, "Technology supports for state science assessment systems," in *Paper Commissioned by the National Research Council Committee on Test Design for K-12 Science Achievement*, The National Academies Press, Washington, DC, USA, 2004.

[7] C. Romero and S. Ventura, "Educational data mining: a survey from 1995 to 2005," *Expert Systems with Applications*, vol. 33, no. 1, pp. 135–146, 2007.

[8] National Science and Technology Council, *The Federal Science, Technology, Engineering, and Mathematics (STEM) Education Portfolio*, Executive Office of the President, Washington, DC, USA, 2011.

[9] J. P. Leighton and M. J. Gierl, "Defining and evaluating models of cognition used in educational measurement to make inferences about examinees' thinking processes," *Educational Measurement: Issues and Practice*, vol. 26, no. 2, pp. 3–16, 2007.

[10] W. A. Mehrens, "Using performance assessment for accountability purposes," *Educational Measurement: Issues and Practice*, vol. 11, no. 1, pp. 3–9, 1992.

[11] J. Brown, S. Hinze, and J. W. Pellegrino, "Technology and formative assessment," in *21st Century Education*, T. Good, Ed., Sage, Thousand Oaks, Calif, USA, 2008.

[12] I. I. Bejar, "Educational diagnostic assessment," *Journal of Educational Measurement*, vol. 21, no. 2, pp. 175–189, 1984.

[13] D. B. Clark, B. Nelson, P. Sengupta, and C. D'Angelo, "Rethinking science learning through digital games and simulations: genres, examples, and evidence," in *Paper Ccommissioned for the National Research Council Workshop on Gaming and Simulations*, Washington, DC, USA, October 2009.

[14] H. Radatz, "Error analysis in Mathematics education," *Journal for Research in Mathematics Education*, vol. 10, no. 3, pp. 163–172, 1979.

[15] E. García, C. Romero, S. Ventura, C. de Castro, and T. Calders, "Association rule mining in learning management systems," in *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. S. J. D. Baker, Eds., pp. 93–106, CRC Press, Boca Raton, Fla, USA, 2011.

[16] C. Romero, P. González, S. Ventura, M. J. del Jesus, and F. Herrera, "Evolutionary algorithms for subgroup discovery in e-learning: a practical application using Moodle data," *Expert Systems with Applications*, vol. 36, no. 2, pp. 1632–1644, 2009.

[17] G. K. W. K. Chung, E. L. Baker, T. P. Vendlinski et al., "Testing instructional design variations in a prototype math game," in *Proceedings of the Annual Meeting of the American Educational Research Association*, Denver, Colo, USA, April 2010.

[18] National Research Council, *Learning Science through Computer Games and Simulations*, The National Academies Press, Washington, DC, USA, 2011.

[19] S. Amershi and C. Conati, "Automatic recognition of learner types in exploratory learning environments," in *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. S. J. D. Baker, Eds., pp. 389–416, CRC Press, Boca Raton, Fla, USA, 2011.

[20] F. Bonchi, F. Giannotti, C. Gozzi et al., "Web log data warehousing and mining for intelligent web caching," *Data and Knowledge Engineering*, vol. 39, no. 2, pp. 165–189, 2001.

[21] W. J. Frawley, G. Piateski-Shapiro, and C. J. Matheus, "Knowledge discovery in databases: an overview," *AI Magazine*, vol. 13, no. 3, pp. 57–70, 1992.

[22] C. Romero, S. Ventura, A. Zafra, and P. D. Bra, "Applying Web usage mining for personalizing hyperlinks in web-based adaptive educational systems," *Computers and Education*, vol. 53, no. 3, pp. 828–840, 2009.

[23] C. Romero, S. Ventura, M. Pechenizkiy, and R. S. Baker, *Handbook of Educational Data Mining*, CRC Press, Boca Raton, Fla, USA, 2001.

[24] G. K. W. K. Chung and D. Kerr, "A primer on data logging to support extraction of meaningful information from educational games: an example from Save Patch," CRESST Report 814, University of California, National Center for Research on Evaluation, Standards, and Student Testing, Los Angeles, Calif, USA, 2012.

[25] R. Berkhin, "A survey of clustering data mining techniques," in *Grouping Multidimensional Data*, J. Kogan, C. Nicholas, and M. Teboulle, Eds., pp. 25–72, Springer, New York, NY, USA, 2006.

[26] F. C. James and C. E. McCulloch, "Multivariate analysis in ecology and systematics: Panacea or Pandora's box?" *Annual Review of Ecology and Systematics*, vol. 21, no. 1, pp. 129–166, 1990.

[27] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data Mining and Knowledge Discovery*, vol. 2, no. 3, pp. 283–304, 1998.

[28] L. A. Roussos, W. F. Stout, and J. I. Marden, "Using new proximity measures with hierarchical cluster analysis to detect multidimensionality," *Journal of Educational Measurement*, vol. 35, no. 1, pp. 1–30, 1998.

[29] A. Vellido, F. Castro, and A. Nebot, "Clustering educational data," in *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. S. J. D. Baker, Eds., pp. 75–92, CRC Press, Boca Raton, Fla, USA, 2011.

[30] W. Vogt and D. Nagel, "Cluster analysis in diagnosis," *Clinical Chemistry*, vol. 38, no. 2, pp. 182–198, 1992.

[31] M. Maechler, "Cluster: cluster analysis extended Rousseeuw et al," R package version 1.14.3, 2012.

[32] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Development Core Team, 2010.

[33] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 4, no. 1, pp. 300–307, 2007.

[34] C. Krier, D. Francois, F. Rossi, and M. Verleysen, "Feature clustering and mutual information for the selection of variables in spectral data," in *Proceedings of the European Symposium on Artificial Neural Networks (ESANN '07)*, pp. 157–162, Bruges, Belgium, April 2007.

[35] E. H. Ruspini, "A new approach to clustering," *Information and Control*, vol. 15, no. 1, pp. 22–32, 1969.

[36] F. Tian, S. Wang, C. Zheng, and Q. Zheng, "Research on e-learner personality grouping based on fuzzy clustering analysis," in *Proceedings of the 12th International Conference on Computer Supported Cooperative Work in Design (CSCWD '08)*, pp. 1035–1040, IEEE, Xi'an, China, April 2008.

[37] D. Kerr and G. K. W. K. Chung, "Identifying key features of student performance in educational video games and simulations through cluster analysis," *Journal of Educational Data Mining*, vol. 4, no. 1, pp. 144–182, 2012.

[38] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. R. Zaane, "Clustering and sequential pattern mining of online collaborative learning data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 6, pp. 759–772, 2009.

[39] R. Srikant and R. Agrawal, "Mining sequential patterns: generalizations and performance improvements," in *Advances in Database Technology—EDBT '96: 5th International Conference on Extending Database Technology Avignon, France, March 25–29, 1996 Proceedings*, vol. 1057 of *Lecture Notes in Computer Science*, pp. 1–17, Springer, Berlin, Germany, 1996.

[40] C. Antunes, "Acquiring background knowledge for intelligent tutoring systems," in *Proceedings of the 1st International Conference on Educational Data Mining (EDM '08)*, R. S. J. D. Baker, T. Barnes, and J. E. Beck, Eds., pp. 18–27, Montreal, Canada, June 2008.

[41] M. Zhou, Y. Xu, J. C. Nesbit, and P. H. Winne, "Sequential pattern analysis of learning logs: methodology and applications," in *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. S. J. D. Baker, Eds., pp. 107–121, CRC Press, Boca Raton, Fla, USA, 2011.

[42] D. Cummins, K. Yacef, and I. Koprinska, "A sequence based recommender system for learning resources," *Australian Journal of Intelligent Information Processing Systems*, vol. 9, no. 2, pp. 49–56, 2006.

[43] A. F. Hadwin, J. C. Nesbit, D. Jamieson-Noel, J. Code, and P. H. Winne, "Examining trace data to explore self-regulated learning," *Metacognition and Learning*, vol. 2, no. 2-3, pp. 107–124, 2007.

[44] C. Buchta and M. Hahsler, "ArulesSequences: Mining frequent sequences," R package version 0.2-4, 2013.

[45] D. Harel, "Statecharts: a visual formalism for complex systems," *Science of Computer Programming*, vol. 8, no. 3, pp. 231–274, 1987.

[46] J. Olive and J. Lobato, "The learning of rational number concepts using technology," in *Research on Technology in the Learning and Teaching of Mathematics*, K. Heid and G. Blume, Eds., pp. 1–53, Information Age Publishing, Greenwich, Conn, USA, 2008.

[47] Franz Inc, Gruff, http://franz.com/agraph/gruff/.