Analyzing Process Data from Game/Scenario-Based Tasks: An Edit Distance Approach

Jiangang Hao
Center for Advanced Psychometrics
Educational Testing Service
Princeton, NJ 08541, U.S.A.

Zhan Shu
Educational Testing Service
Princeton, NJ 08541, U.S.A.

Alina von Davier
Center for Advanced Psychometrics
Educational Testing Service
Princeton, NJ 08541, U.S.A.

January 6, 2015

Students' activities in game/scenario-based tasks (G/SBTs) can be characterized by a sequence of time-stamped actions of different types with different attributes. For a subset of G/SBTs in which only the order of the actions is of great interest, the process data can be well characterized as a string of characters (i.e., action string) if we encode each action name as a single character. In this article, we report our work on evaluating students' performances by comparing how far their action strings are from the action string that corresponds to the best performance, where the proximity is quantified by the edit distance between the strings. Specifically, we choose the Levenshtein distance, which is defined as the minimum number of insertions, deletions, and replacements needed to convert one character string into another. Our results show a strong correlation between the edit distances and the scores obtained from the scoring rubrics of the pump repair task from the National Assessment of Education Progress Technology and Engineering Literacy assessments, implying that the edit distance to the best performance sequence can be considered as a new feature variable that encodes information about students' proficiency, which sheds light on the value of data-driven scoring rules for test and task development and for refining the scoring rubrics.

1. INTRODUCTION

Innovations in educational assessments have been accelerated by the advance of technology over the past decade. The objective of educational assessments is to collect and make sense of information about what students know and can do and to evaluate their progress or shape their

future learning experience (Mislevy et al., 2014). The more students are engaged and put effort into the assessments, the better will be the information collected about them (Schmit and Ryan, 1992; Sundre and Wise, 2003). Gee (2007) has pointed out that video games or simulations have the capability to increase students' engagement and to create better conditions that boost learning. From the assessment perspective, game/scenario-based tasks (G/SBTs) promise to offer a sweet spot in the assessment design space for some purposes and some circumstances (Mislevy et al., 2014). Moreover, G/SBTs can provide students with new opportunities to demonstrate proficiencies in complex interactive environments that traditional assessment formats cannot afford (Klopfer et al., 2009).

Given these nice features, G/SBTs are considered one of the major future directions of educational assessment and have received considerable attention over the past decade. In practice, G/SBTs have been widely used in real assessments ranging from low- to medium-stakes tests, such as the Technology and Engineering Literacy assessments (TEL) from the National Assessment of Educational Progress (NAEP) (TEL, 2013), to high-stakes tests, such as the innovative assessments in the U.S. Medical Licensure Examinations (USMLE, 2014).

For traditional item formats, such as multiple choice (MC) or constructed response (CR), the scoring rubrics are generally straightforward to develop and implement. But for G/SBTs, developing the scoring rubrics itself becomes much more complicated, and it is no longer a trivial task to implement the scoring rubrics too. The complications come from at least two sources. First, owing to the increased complexity provided by the interactive environment of G/SBTs, developing operational scoring rubrics requires more iterations with the help of proper data-mining techniques to uncover meaningful features from students' activities. G/SBTs are generally developed by following an evidence-centered design (ECD) (Almond et al., 2002; Mislevy and Riconscente, 2006), where several rounds of iterations are implemented to explicate the actions or state information relevant to the targeted construct. However, in practice, it is almost impossible to predict all possible behaviors in the game or simulation ahead of time, which makes data mining the log file a necessary step to uncovering empirical evidence relevant to the construct to be measured. Second, the logistics of handling the log files from the G/SBTs are very challenging. Students' activities are kept in the log files, and parsing through the log file to extract useful information is generally not a trivial task, requiring additional data analysis techniques.

Discovering new features from the logs of specific G/SBTs requires creativity and insight, which are generally difficult to standardize. Therefore, finding new approaches that can be applied to rather generic G/SBTs is highly desirable. There are a lot efforts have been devoted to analyzing the sequential data from educational games/simulations. For example, temporal Bayesian Networks has been used to model students' performance in Orthopedic Surgery Training (Chieu et al., 2010). Various sequence detection techniques are discussed for generating adaptive feedback in mathematical generalisation (Gutierrez-Santos et al., 2010). Cluster analysis has been used to analyze action sequences in personalized e-learning (Köck and Paramythis, 2011), scenario based assessment (Bergner et al., 2014) and systematic inquiry behaviors (Sao Pedro et al., 2013). In addition to directly cluster the event sequence, the clustering of activity state sequence with interval sampling technique has been studied (Desmarais and Lemieux, 2013). Moreover, a set of tools used for sequence mining have been assembled together into a R package, TraMineR (Gabadinho et al., 2009).

In this article, we propose an edit distance–based approach to extract useful information

from the logs of certain types of G/SBTs, where we know what the best practices are.[1] In this approach, students' performances are measured by how far or close their action sequences are from the action sequences corresponding to the "best" practices. To measure how far or close they are, we propose to use the "action distance," which is the (weighted) number of certain operations needed to bring one action sequence to another (usually, the one corresponding to the best practice). If we dummy code each action name as a single character and choose the operations as some text-editing operations, such as insertion, deletion, and substitution, the action distance is exactly the well-known edit distance that originally emerged in natural language processing (NLP).

To demonstrate the usefulness of this approach, we apply it to a specific task, the pump repair task from the NAEP TEL (PumpRepair, 2013). In our analysis, we dummy code the action names with lowercase letters and turn each student's response into a character string. Then, to compare the differences between the strings, we choose one of the most widely used edit distances, the Levenshtein distance (Levenshtein, 1966), which is defined as the minimum number of deletions, insertions, and substitutions that will transform one string into another. Our analysis shows that though such a measure is obtained from a very different perspective, it correlates significantly with the scores obtained from the scoring rubrics. This provides a new perspective from which we can quantify students' performances in G/SBTs and that can be readily applied to a number of similar G/SBTs where only the order of the actions is of primary interest.

It is worth noting that the approach presented in this article, like other data-mining approaches, should not be considered as a replacement for rubric-based scoring. Rather, data-mining approaches should be considered independent checks for scoring rubrics, useful in flagging potential problems. Direct score reporting based on data-mining approaches can only be done after results are properly interpreted.

The article is organized as follows. In section 2, we introduce edit distance, the basic algorithms, and their applications to G/SBAs. In section 3, we apply the edit distance approach to a specific simulation-based task, the pump repair task from NAEP, and show the results. In section 4, we discuss the applicabilities and limitations of the edit distance approach.

2. EDIT DISTANCE

2.1. FROM ACTION DISTANCE TO EDIT DISTANCE

A student's response to the G/SBTs forms a sequence of time-stamped actions, which we refer to as an action sequence. For certain types of G/SBTs, we know what action sequences correspond to the best performance based on the design rubrics, which we refer to as ideal action sequences. Therefore, we can introduce an action distance that is a measure of how far the action sequence from a response is to the ideal action sequence. The action distance reduces to the edit distance if we dummy code each action name as a single character, ignoring the temporal component corresponding to the time stamp of each action and restricting the operations only to editing operations such as insertion, deletion, and substitution. In this article, we focus on a subset

[1] Note that, in many cases, finding the best practice is one of the important goals of data mining. Here we assume we know this, which is true for a subset of G/SBTs.

of the G/SBTs where the temporal information is not of major concern.[2] Therefore, from now on, we simply use edit distance as a surrogate for action distance. Edit distance is defined as the minimum weights (costs) of the editing operations used to transform one string into another (Jurafsky and Martin, 2000). Under this definition, there can be many possible variants in terms of the types of editing operations (such as insertion, deletion, substitution, transposition) and the associated weight of each operation.

It is worth noting that we are not fully free to use any type of operation and weight. To properly define the edit distance in metric space, some rules (known as metric axioms) must be met (Bryant, 1985). Assume that we have two strings, denoted as $X[1, ...i, ..N]$ and $Y[1, ...j, ...M]$; the additional requirements are depicted by

- $d(X, Y) \geq 0$ if $X \neq Y$: Nonnegativity
- $d(X, Y) = 0$ if $X = Y$: Identity of the indiscernible
- $d(X, Y) = d(Y, X)$: Symmetry
- $d(X, Y) \leq d(X, Z) + d(Z, Y)$: Triangle inequality

where $d(X, Y)$ denotes the edit distance between string X and string Y. These conditions lead to the following requirements for the operations and weights: (1) there is always an inversion operation with equal weight for each editing operation and (2) all weights associated with the edit operations are positive. These requirements significantly reduce the possible space of the variants of the edit distance and are important guidelines for "inventing" new edit distances.

Obviously, the choice of specific edit distance may be "optimized" for a specific purpose by adjusting the types of edit operations and their associated weights. However, for exploratory data analysis, where we generally do not have information about what is the best way to adjust those parameters a priori, it will be sensible to start with the simplest situation, that is, the Levenshtein distance. Throughout the rest of the article, unless noted otherwise, all edit distances we discuss are Levenshtein distances.

2.2. Weighted Levenshtein distance

Among all possible variants of the edit distance, the most widely used is the Levenshtein distance. If we keep the edit operations but allow the weights associated with each operation to be variable, we arrive at the weighted Levenshtein distance (Jurafsky and Martin, 2000). This is a more general situation than the Levenshtein distance, and manipulating the weights gives one an extra knob to "optimize" the edit distance for a specific task. So, in the following, we introduce the algorithm for the weighted Levenshtein distance and hold the Levenshtein distance as a special case where all weights are set to 1. Algorithmically, the computational time for calculating the weighted Levenshtein distance is $O(M * N)$ when realized by dynamic programming (Jurafsky and Martin, 2000). The algorithm is generally specified as two steps. The first step is initialization:

[2]For a subset of G/SBTs, the order of the actions encodes most of the information, and we can ignore the specific temporal information of each action.

$$
\begin{aligned}
d_{00} &= 0, \\
d_{i0} &= d_{i-1,0} + w_{del}(X_i), \qquad 1 \le i \le N, \\
d_{0j} &= d_{0,j-1} + w_{ins}(Y_j), \qquad 1 \le j \le M.
\end{aligned} \tag{1}
$$

Following initialization, a recurrence relation updates the edit distance:

$$
d_{ij} = \begin{cases} d_{i-1,j-1}, & X_i = Y_j, \\ \min \begin{cases} d_{i-1,j} + w_{\text{del}}(X_i), \\ d_{i,j-1} + w_{\text{ins}}(Y_j), \\ d_{i-1,j-1} + w_{\text{sub}}(Y_j, X_i). \end{cases} & X_i \neq Y_j, \end{cases} \tag{2}
$$

where the functions w_{del}, w_{ins}, and w_{sub} denote the weights of deletion, insertion, and substitution, respectively. Note that the weights are not necessarily constants for all i and j. The edit distance between X and Y is given by d_{NM}. As we mentioned earlier, adjusting the weights provides an extra handle on the resulting edit distance. When there are well-motivated reasons to choose a specific set of weights, one should go with it. For example, the keyboard layout makes certain typos more likely than others. Depending the purpose of the analysis, one can increase or decrease the weights associated with the editing operations relevant to those keys. But in general situations, such as the application discussed in this article, we usually do not have a clear motivation for a specific choice for the weights. From a Bayesian point of view, such weight choices can be considered a priori based on the edit distance.

2.3. Application to G/SBTs

The process data of a student's response to a G/SBT can be summarized as a sequence of time-stamped actions with corresponding attributes. Depending on the complexity of the specific game or scenario-based task, this sequence of actions can be very different in terms of length and variability. For a subset of G/SBTs, the time stamp plays a minor role, as the student's performance is mainly determined by the order of his or her actions during the task. Then, each student's performance is fully characterized by a sequence of actions whose order encodes the majority of information about the performance. As we show, the edit distance approach is well suited for this subset of G/SBTs.

For certain types of G/SBTs, we know the action sequences corresponding to the best practices, that is, the ideal action sequences. The ideal action sequences can be a single sequence from the beginning to the end of the task or a set of segmented sequences, each of which corresponds to evidence nuggets that support the proficiency of a certain skill. Conversely, the action sequences from students' actual responses can be very diverse, some of them following closely or being exactly the same as the ideal action sequences, whereas others may be very different. Therefore, how close the actual response is to the ideal action sequence provides a measure of how good the student's performance is, if one can appropriately specify the metric for "closeness." We suggest that edit distance, specifically the Levenshtein distance, can be considered a promising candidate for this purpose.

Now, the most challenging question comes. How do we quantify that the edit distance does its job properly? Is there a method that allows us to test whether the edit distance for a specific

task really works properly? The answer is yes. What we propose is to compare edit distances calculated based on students' actual responses to the edit distances calculated based on fake responses after randomizing the orders of the actions. If the distribution of edit distances from actual responses is systematically lower than the distribution from random responses, it is a good indication that the edit distance is doing its job decently. If the distribution of the edit distance from actual responses is statistically indistinguishable from the distribution calculated based on random responses, then the edit distance won't provide much useful information. This also gives a way to winnow good ideal action sequences that can be used to calculate the edit distance.

To apply the edit distance as a measure of proximity between actual responses and ideal action sequences, another prerequisite is easily neglected. That is, we need to dummy code action names into single letters or numbers because the edit distance is operating directly on character strings. For the dummy coding, there are at least two different situations. In the first situation, one just codes each action as a unique single letter or number. In the second situation, instead of coding each action as a unique letter or number, one codes a class of actions to the same unique letter or number; that is, the actions can be classified into different groups first, and then the dummy coding is done at the group level. This has important implications in practice, because it will be possible that there are actions that play similar roles in terms of revealing the proficiency of certain constructs, and one may want to treat all these actions the same.

Last, but not least, if a task has multiple ideal action sequences (sequence segments), extra wisdom is needed to decide how to combine the edit distances corresponding to these segments. A confirmatory factor analysis can check an optimal concatenation of the edit distances if there is a clear cognitive motivation. If there is no clear motivation, principal component analysis (PCA) or cluster analysis can guide the combination. In the next section, we demonstrate all these ideas using a specific scenario-based task: the pump repair task from the NAEP TEL.

3. PUMP REPAIR TASK

3.1. THE TASK AND SCORING RUBRICS

The TEL (TEL, 2013) is trying to measure whether students can apply what they learn about technology and engineering skills to real-life situations. TEL is implemented via "computer-based and interactive scenario-based tasks to gauge what students know and can do." Among the TEL tasks, the pump repair task was released to the public after the TEL pilot study was conducted in 2013. In the pump repair task, students are asked to play the role of an engineer to troubleshoot a water well that fails to work in a remote village in Nepal. There are two major parts in the pump repair task. In the first part, students can ask a set of questions about why the pump does not work and can get hints about the causes of the problems. In the second part, the students set out to fix the problem. According to the "pump manual," students are told that there are five common problems for the hand pump. The recommended cycle for troubleshooting each problem is check first, followed by repair, and then followed by a test to determine whether it is fixed. In Figure 1, we include two screenshots from the pump repair task, which correspond to the two major phases of the task.

Students' responses are recorded in the following way: the check, repair, and test operations are denoted by C, R, and P, respectively. The five common problems are indexed as 1 through 5. For example, if a student performs the operations of check the fourth common problem followed by repairing the pump for that problem, followed by testing the pump, his or her action sequences

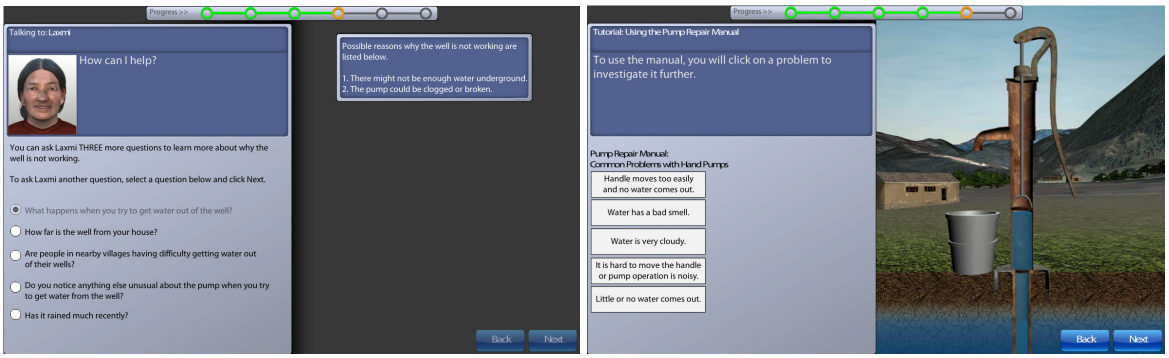

Figure 1: Screenshots of the pump repair task. (left) Questions students can ask the system to get hints about problems with the well system. (right) Troubleshooting part of the task, where students can choose to fix the problems.

Table 1: The scoring rules along the systematicity dimension

Score Level	Details
4	All checks performed before repairs Pump is checked immediately following each repair
3	All checks performed before repairs Pump is not checked immediately following each repair
2	One repair is performed before the associated check (or check is omitted) Pump may not be checked immediately following each repair
1	Two or more repairs performed before the associated check Pump may not be checked immediately following each repair

will be recorded as C4R4P. The scoring rubrics used in the pilot study defined two dimensions for the construct probed by this pump repair task: systematicity and efficiency. Systematicity refers to the idea that students need to follow a systematic operation cycle, that is, check, repair, and test, when they troubleshoot the well. Efficiency refers to the idea that students can quickly identify which problems the well actually has and repair them. According to the design of the task, if students are very careful in the first part of the task, they should be able to infer that only problems 4 and 5 are real trouble makers and need to be checked, fixed, and tested.

Each student's response is recorded in the log files as we described earlier. The scoring rubric (TEL, 2013) of the task defines the scoring rules for each of the two dimensions, as follows. For systematicity, the performances are classified into four levels, as shown in Table 1. Conversely, for efficiency, there are five levels, as shown in Table 2.

On the basis of the scoring rubrics, the responses from a total of 1,325 students in the pilot study are scored; a cross table of the score distribution is shown in Figure 2. Looking at the distribution, the two dimensions have low correlation, with a Spearman correlation of $r = 0.347$. Notably, one can observe from Figure 2 that 162 students got the highest level of efficiency score but the lowest level of systematicity score. Conversely, only four students are placed at the highest level of systematicity score but get the lowest efficiency score. This means that many students know where the problems are but cannot fix them systematically, but fewer people who can fix the problem systematically do not know where the problems are.

Table 2: The scoring rules along the efficiency dimension

Action definitions	Efficient actions: E = P, C4, R4, C5, R5 Unnecessary checks: C = C1, C2, C3 Unnecessary repair: R = R1, R2, R3
Score Level	**Details**
5	Only actions from set E
4	Actions from E + 1 action from C
3A	Actions from E + 2–3 actions from C
3B	Actions from E + 0–1 action from C + 1 action from R
2	Actions from E + 2–3 actions from C + 1–2 actions from R
1	Actions from E + 3 actions from C + 3 actions from R

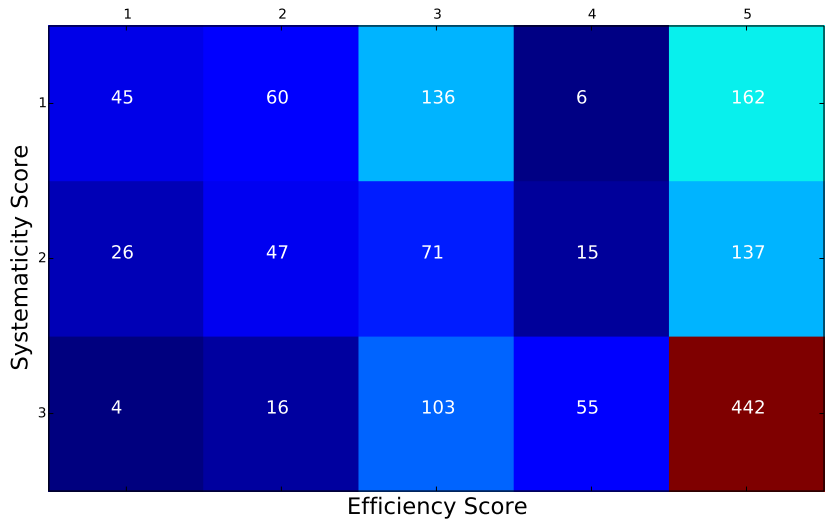

Figure 2: Cross table of the efficiency score and systematicity score for the responses from the pilot study. Note that for the systematicity score, no responses get to level 4.

Table 3: Two dummy coding schemes

Action name	C1	C2	C3	C4	C5	R1	R2	R3	R4	R5	P
Coding scheme I	a	b	c	d	e	f	g	h	i	j	k
Coding scheme II	a	a	a	b	b	c	c	c	d	d	e

3.2. DUMMY CODING OF ACTION NAMES

Students' actions are recorded, for example, as "C4, R4, P, C5, R5, P." To apply the edit distance analysis, one needs to turn this into a string with each action recoded as a single letter; that is, one needs to recode the preceding response string by mapping each action or some actions to a single letter. In our implementation, we choose two coding schemes. In coding scheme I, we just convert each action name into a single and unique letter. In coding scheme II, we first classify the actions as necessary actions if they are associated with either 4 or 5 or P. The rest will be classified as unnecessary actions. Then, we code all the unnecessary actions of the same type (e.g., check, repair) to the same letter. The reason for considering coding scheme II is that we want to see whether the edit distance measure will lead to different results if we dummy code the actions in a different but meaningful way. In Table 3, we list the specific mappings of the two schemes.

It is worth noting that the way one chooses to code the action names is somewhat arbitrary in a certain sense. But comparing with random responses for each coding scheme will provide a way to test whether the coding scheme is reasonable.

3.3. EDIT DISTANCE OF THE RESPONSES

In this subsection, we present the results of the edit distances calculated with respect to the action sequences. For the pump repair task, we know the action sequences that correspond to the highest level of skill, for example, C4R5PC5R5P and C5R5PC4R4P. As these two ideal sequences are equally good, we will choose a final edit distance that is the minimum of the edit distances corresponding to each of them. In addition to the ideal action sequences, some other action sequence segments represent certain levels of system thinking, for example, C1R1P, C2R2P, C3R3P, C4R4P, and C5R5P. So we also calculate the edit distances between the actual responses and these action sequence segments. By doing this, we hope to capture all possible information via the edit distances and then explore the space these edit distances span via PCA and cluster analysis. To facilitate the discussion, we introduce the abbreviations corresponding to different edit distances in Table 4.

The first thing we want to check is a comparison of edit distances from actual responses to those from random responses. We create the random responses in the following way: after coding the actual responses into character strings, we build a set of string lengths by counting the length of each character string. Then, we randomly sample the letters based on the coding schemes to form random strings whose lengths are randomly sampled from the set of character string lengths. According to the task design, one must have the test operation (P) done before he or she can submit the results. So, when we sample the letters, we reserve the last letter of each string as that corresponding to action P. Through this process, we generated two sets of random responses corresponding to the two coding schemes. Then, we calculated the edit distances based on these random responses in the same way we did for the actual responses. In

Table 4: Abbreviations for edit distances corresponding to different action sequences

Abbreviation	Action sequence
dist0	C4R4PC5R5P
dist1	C5R5PC4R4P
dist2	C4R4P
dist3	C5R5P
dist4	C1R1P
dist5	C2R2P
dist6	C3R3P
dist	min(dist0, dist1)

Figure 3, we plot the distributions of the edit distances from both actual responses and random responses under the two coding schemes. Looking at these figures, one can observe that only the edit distances "dist" from actual responses are significantly less than those from the random responses; "dist2" and "dist3," from actual responses, are slightly less than those from random responses, whereas for the "dist4," "dist5," and "dist6," there are no clear differences between actual responses and random responses. This suggests that "dist" contains most of the useful information about students' performances.

The second thing we want to check is how well the edit distances are associated with the systematicity and efficiency scores defined in the scoring rubrics. Here we mention association rather than correlation because the edit distances' relation may not be linear, whereas correlation mainly captures the linear association. We calculate both the Spearman correlation and the adjusted mutual information (Vinh et al., 2009) between the scores and the edit distances. The results are presented in Figure 4. On the basis of the results, one can observe that the trends of the associations for the Spearman correlation and adjusted mutual information are very close. To make the discussion more intuitive, we focus on the Spearman correlation in the following discussion. For the Spearman correlation, coding scheme I leads to higher absolute correlations. All edit distances show relatively high correlations with the efficiency score, and the highest correlation is from the "dist," which is about -0.82. For the systematicity score, only "dist" shows a significant correlation of -0.56, while the other edit distances do not correlate to the systematicity score significantly. Similar conclusions held true based on the adjusted mutual information. Given that "dist" contains most of the information, it will be interesting to see the cross tables between "dist" and the efficiency and systematicity scores for the two coding schemes. We present the results for both coding schemes in Figure 5.

By carefully examining all these results, the following conclusions can be drawn. First, the two different coding schemes do not lead to significantly different edit distances in terms of the correlations and adjusted mutual information with the systematicity and efficiency scores, though some variabilities have been introduced. Clearly one can also introduce other coding schemes based on specific motivations and compare the results in a similar fashion. In this article, our goal is to demonstrate the methodology rather than give an exhaustive analysis of various coding schemes under different motivations, so we conclude our discussion along this line. Second, the edit distance to the ideal action sequences encodes the most information about students' performance, as revealed both by correlations and adjusted mutual information with the rubric-based scores and by the distribution comparison with random responses. This in-

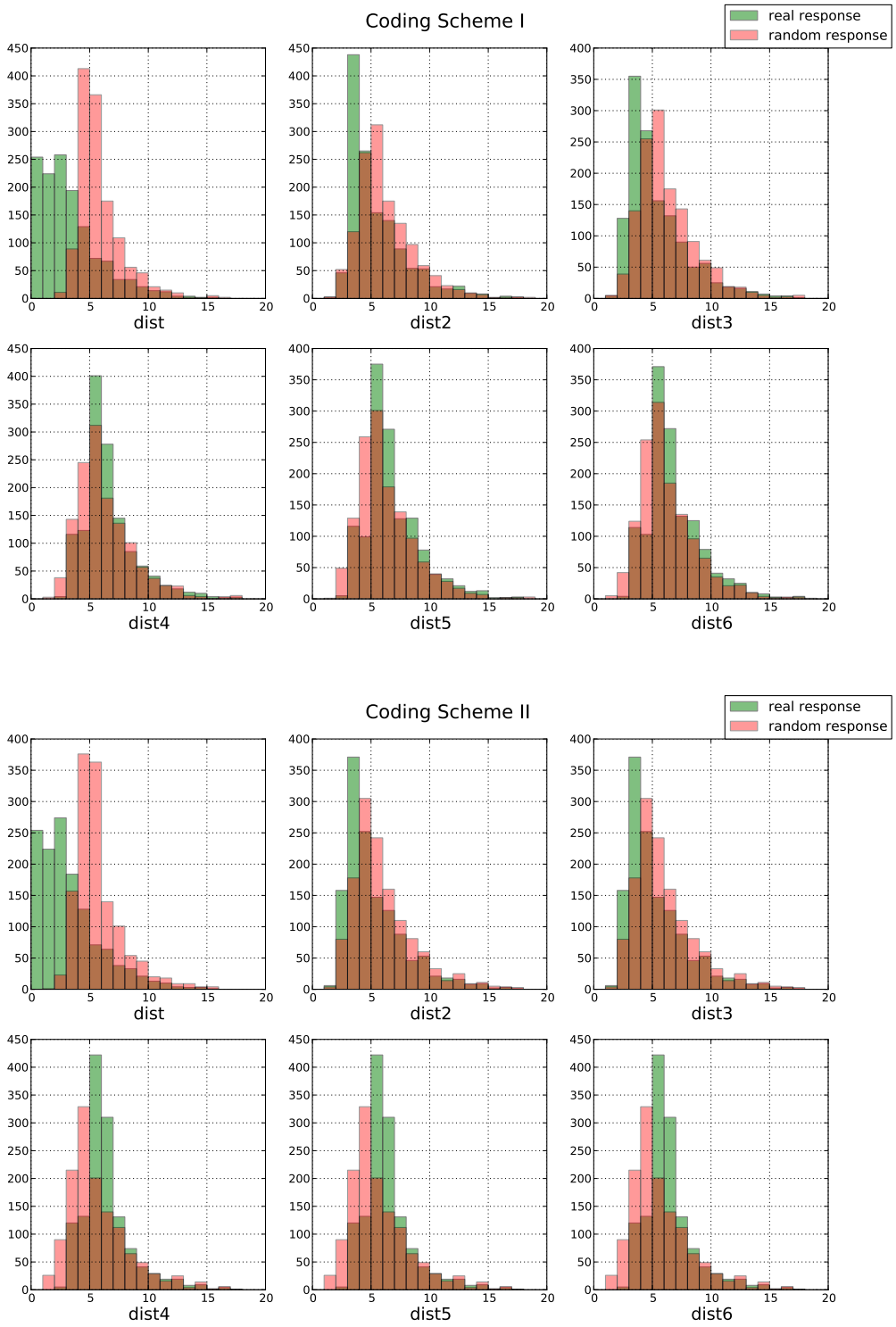

Figure 3: Distributions of the edit distances from actual responses and from random responses for coding schemes I and II. The specific definitions of the labels along the x axis are given in Table 4.

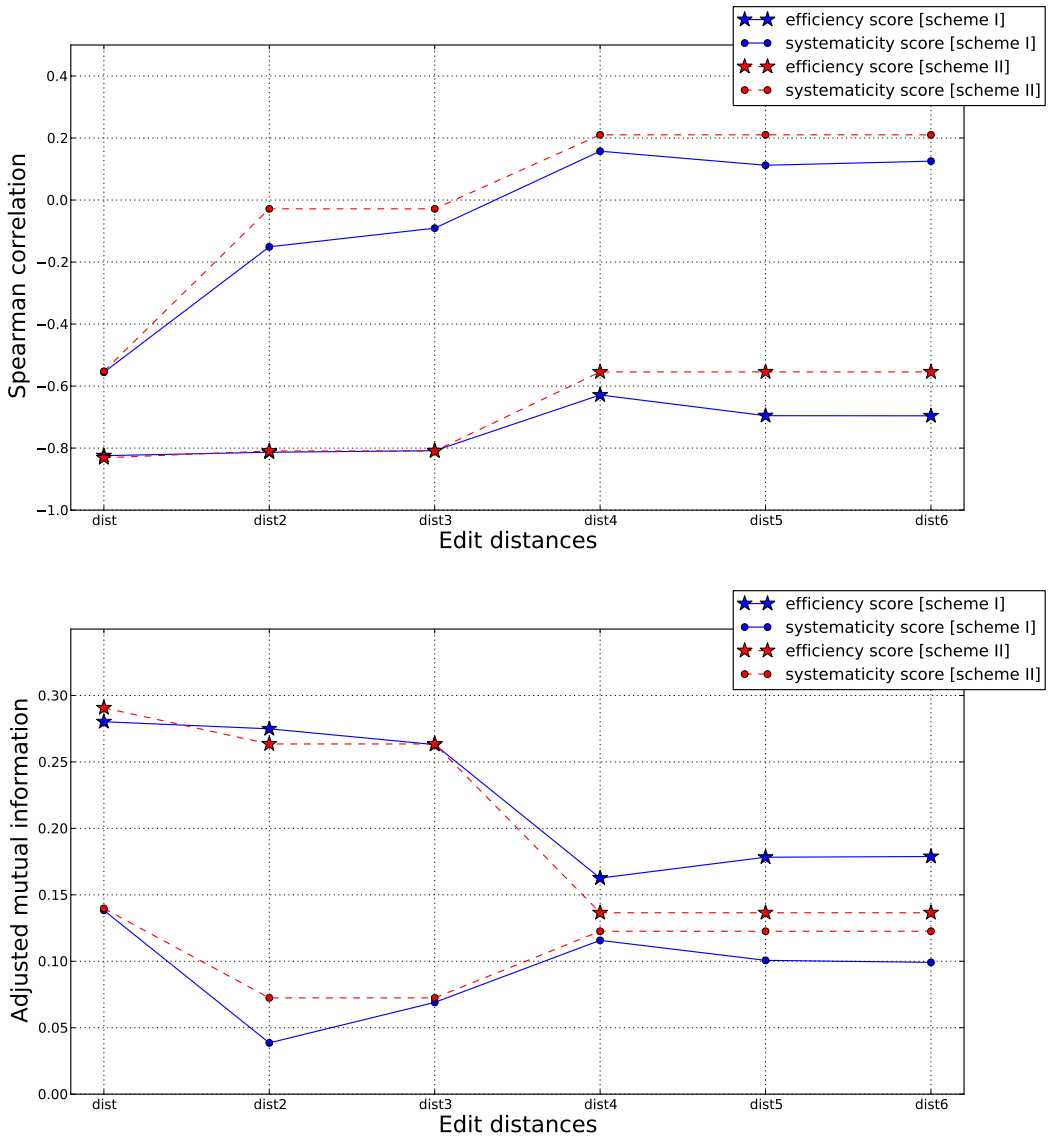

Figure 4: (top) Spearman correlation between the edit distances and the systematicity and efficiency scores from the scoring rubrics. (bottom) Adjusted mutual information between the edit distances and the systematicity and efficiency scores.

forms us that the edit distances to the ideal action sequences are the best first guess to create an edit distance–based measure to quantify students' performance in G/SBTs, and thus can serve as an informative feature variable that characterizes the performance. Third, most of the edit distances other than "dist" do not display distinct distributions compared with those calculated from random responses, whereas most of them do show high correlations with the efficiency score; this implies that the efficiency score may not be a well-defined dimension of the construct as it highly correlates with those edit distances that do not manifest clear distinctions between real responses and random responses. Finally, the information encoded in the edit distances is different from the information contained in the rubric-based scores, though there are reasonably high (negative) correlations. Each of these approaches probes the process data from a specific perspective, and the vector space all these features span needs to be analyzed to extract complete information about the students' performances in G/SBTs. In the next subsection, we focus on the vector space spanned by the edit distances corresponding to different action sequences and explore the ways to combine them.

3.4. Principal component analysis and hierarchical cluster analysis

The various edit distances corresponding to different action sequences lead to a multidimensional (six, in our case) vector space. Given the possible correlations among the edit distances, a natural question is, what is the actual dimension of this space, and what are the ways to combine these edit distances that are on the same dimension? To tell how many actual dimensions the vector space really has, probably the best approach is to perform a PCA. In Figure 6, we show results from the PCA for the two coding schemes. The results show that both coding schemes lead to two effective dimensions (PCs) based on the scree plot for the vector spaces spanned by the six edit distances. The first PC is predominant, explaining over 90% of the variability of the data.

It is interesting to see how these PCs correlate with the scores from the scoring rubrics, because the results will tell us whether the space spanned by the edit distances covers the space spanned by the systematicity and efficiency scores. We show the results of the correlation in Figure 7. One can see that the PCs from the two coding schemes correlate differently with the two rubric-based scores. For coding scheme I, the first PC highly correlates (positively) with the efficiency score, while the second PC highly correlates (negatively) with the systematicity score. The other PCs are not strongly correlated with either score. Similar conclusions are applied to coding scheme II, though with some adjustments. For a more intuitive picture about what the correlation numbers mean, we present the scatter plots between the two rubric-based scores and the first and second PCs corresponding to coding scheme I in Figure 8. All these results show that the space spanned by the edit distances covers the space spanned by the rubric-based scores. Moreover, given that the first PC explains most of the variability in the data, and it is highly correlated with the efficiency score, one can infer that among the two rubric-based scores, the efficiency score is subject to higher variability and therefore might not be a very well defined dimension of the construct. This observation echoes our previous conclusion.

Conversely, one may want to check how we should group the edit distances. If there is a strong cognitive motivation, we can do a confirmatory factor analysis to check the grouping. However, in our case, the cognitive motivation is not yet very clear, and therefore we perform a hierarchical cluster analysis on the edit distances to see whether the results are consistent with our general understanding of the task. In Figure 9, we show the results for the two different

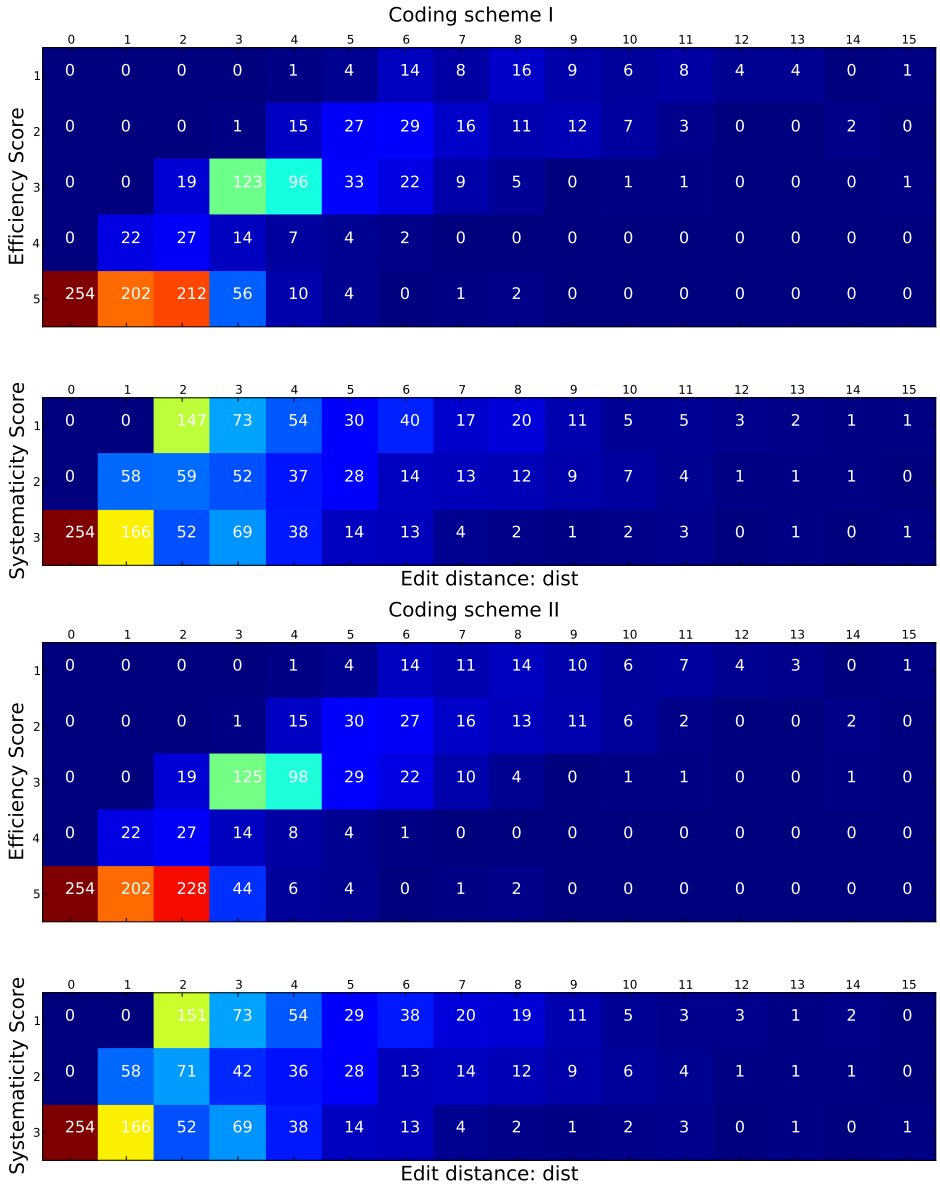

Figure 5: Cross table of the scores from the scoring rubrics vs. the edit distance for coding schemes I and II. Here the edit distance refers to the minimum of the edit distance with respect to C4R4PC5R5P and C5R5PC4R4P.

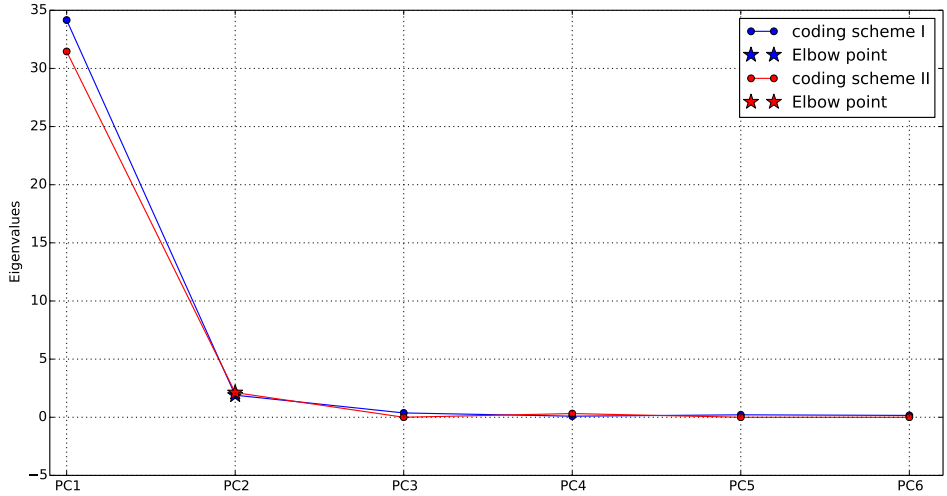

Figure 6: PCA on the space spanned by dist, dist2, dist3, dist4, dist5, and dist6. The results show that different coding schemes lead to slightly different results. Both PCAs indicate that the spaces spanned by the edit distances have two dimensions based on the scree test.

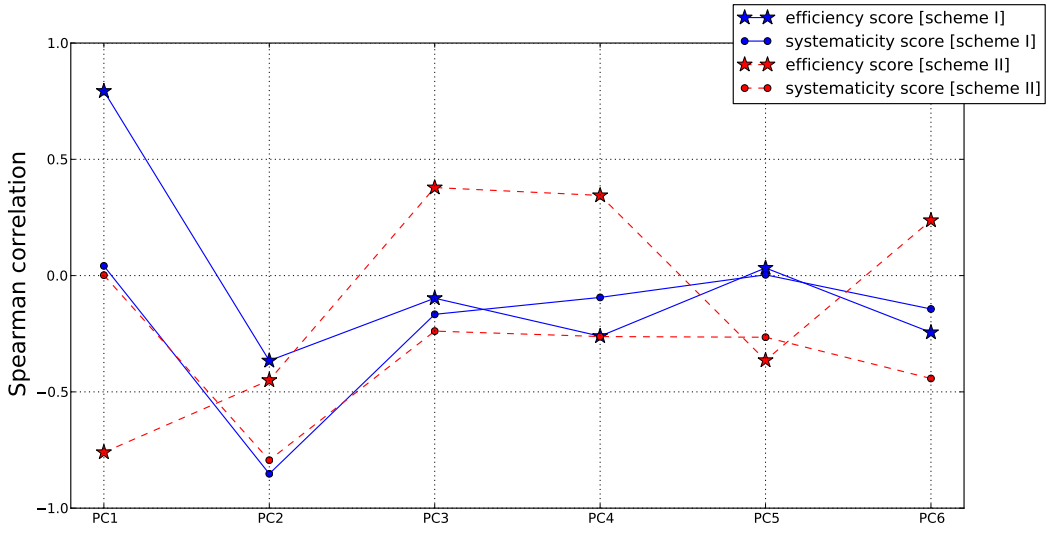

Figure 7: Spearman correlation between the PCs and the systematicity and efficiency scores.

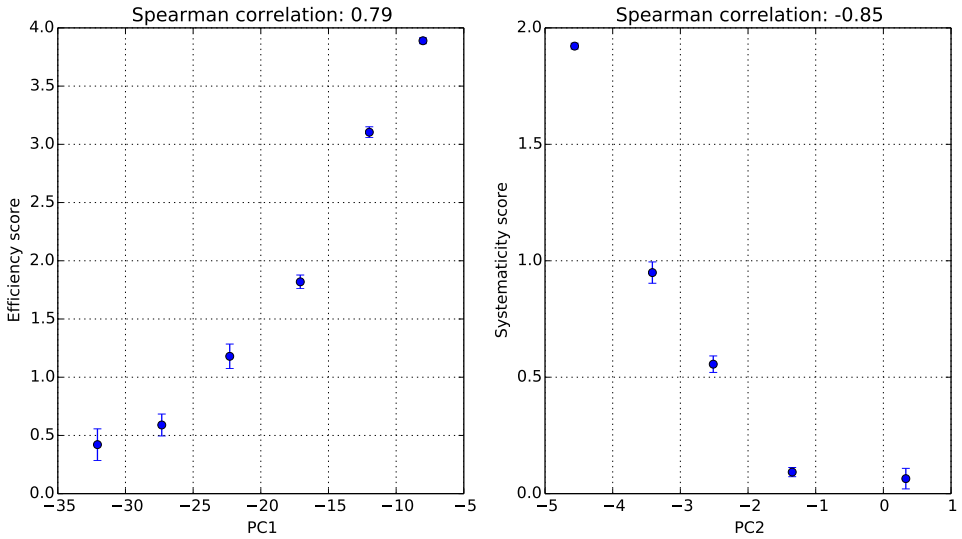

Figure 8: Scatter plots between the first two PCs and the rubric-based scores for coding scheme I. Each dot is the mean value of the bin, and the error bar is the standard deviation of the mean.

coding schemes. In coding scheme I, one can observe that there are three clusters: dist; dist2, dist3; and dist4, dist5, dist6. Such a clustering pattern is very consistent with our understanding of the task. The "dist" corresponds to the ideal action sequence and should be different from others; "dist2" and "dist3" correspond to the main action sequence segments in the ideal action sequence and is different from the rest (e.g., "dist4," "dist5," and "dist6"). So, a simple hierarchical cluster analysis further confirms the consistency of our choice of the ideal action sequences and action sequence segments. Similar conclusions are held for coding scheme II.

4. Discussion

In this article, we propose an edit distance approach to analyzing the process data from certain G/SBTs, where only the order of the actions is of primary interest. By considering a specific task, the pump repair task, from NAEP TEL, we lay down step-by-step procedures for applying the edit distance approach to the process data. By comparing the edit distances with the scores from the existing scoring rubrics of the pump repair task, we conclude that the information contained in the two scores is also reflected by the edit distances, though the latter is obtained from quite a different perspective. Moreover, PCA on edit distances, together with the internal consistency among the edit distances from actual responses and from random responses, suggests that the efficiency score based on the existing scoring rubrics needs further scrutinization.

However, the edit distance approach in its current form does have restrictions on what types of tasks it is applicable to. One of the major restrictions is that the temporal information as well as the properties of different actions are not included in the current scheme. This additional information about the process data may be accommodated into the edit distance scheme by assigning appropriate weights to the edit operations based on the time interval or action attributes. In addition to this restriction, how to properly interpret the "edit" operations is also challenging.

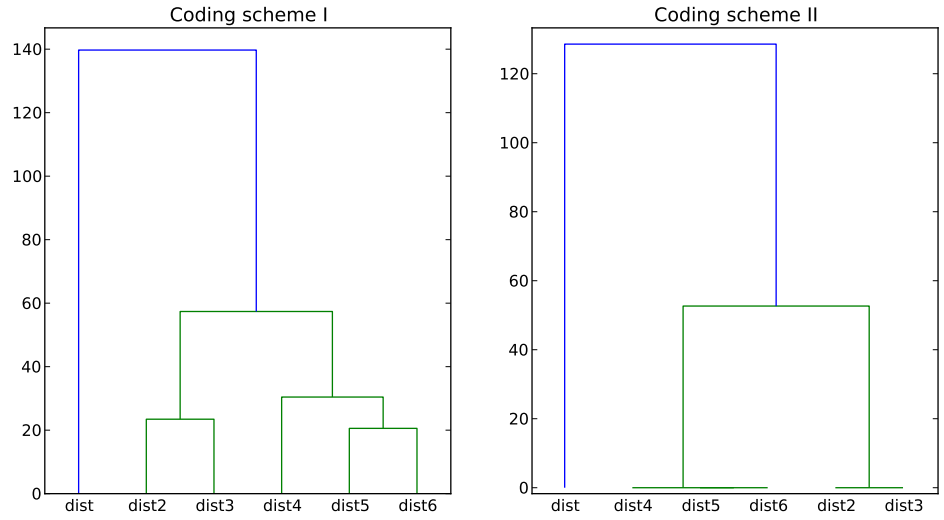

Figure 9: Hierarchical clustering analysis using complete linkage for edit distances dist, dist2, dist3, dist4, dist5, dist6.

The interpretation may be different for different G/SBTs, which leaves great room for imagination. In an ongoing study, we are applying the approach to a number of other G/SBTs, and we hope we can find more generalizable ways to interpret the "edit" operations after comparing across different tasks. Nevertheless, the edit distance approach remains a useful tool for exploring the process data from G/SBTs, allowing quick construction of meaningful feature variables to help to refine the scoring rules.

ACKNOWLEDGMENT

The work is supported by allocation funding from Educational Testing Service. J.H. thanks Shelby Haberman for helpful conversations.

REFERENCES

ALMOND, R., STEINBERG, L., AND MISLEVY, R. 2002. Enhancing the design and delivery of assessment systems: A four-process architecture. *The Journal of Technology, Learning and Assessment 1,* 5.

BERGNER, Y., SHU, Z., AND VON DAVIER, A. 2014. Visualizing and clustering sequence data from a simulation-based assessment task. *Journal of Educational Data Mining*.

BRYANT, V. 1985. *Metric spaces: iteration and application*. Cambridge University Press.

CHIEU, V. M., LUENGO, V., VADCARD, L., AND TONETTI, J. 2010. Student modeling in orthopedic surgery training: Exploiting symbiosis between temporal bayesian networks and fine-grained didactic analysis. *International Journal of Artificial Intelligence in Education 20,* 3, 269–301.

DESMARAIS, M. C. AND LEMIEUX, F. 2013. Clustering and visualizing study state sequences. In *Proceedings of 6th International Conference on Educational Data Mining*, pp. 224–227.

GABADINHO, A., RITSCHARD, G., STUDER, M., AND MÜLLER, N. S. 2009. Mining sequence data in r with the traminer package: A users guide for version 1.2. *Geneva: University of Geneva*.

GEE, J. P. 2007. *What video games have to teach us about learning and literacy.: Revised and Updated Edition*. Macmillan.

GUTIERREZ-SANTOS, S., MAVRIKIS, M., AND MAGOULAS, G. 2010. Sequence detection for adaptive feedback generation in an exploratory environment for mathematical generalisation. In *Artificial Intelligence: Methodology, Systems, and Applications*, pp. 181–190. Springer.

JURAFSKY, D. AND MARTIN, J. H. 2000. *Speech & Language Processing*. Pearson Education India.

KLOPFER, E., OSTERWEIL, S., GROFF, J., AND HAAS, J. 2009. Using the technology of today, in the classroom today. *The Education arcade*.

KÖCK, M. AND PARAMYTHIS, A. 2011. Activity sequence modelling and dynamic clustering for personalized e-learning. *User Modeling and User-Adapted Interaction 21,* 1-2, 51–97.

LEVENSHTEIN, V. I. 1966. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, Volume 10, pp. 707.

MISLEVY, R., ORANJE, A., BAUER, M. I., VON DAVIER, A. A., HAO, J., CORRIGAN, S., HOFFMAN, E., DICERBO, K., AND JOHN, M. 2014. *Psychometric considerations in game based assessments*. CreateSpace Independent Publishing Platform.

MISLEVY, R. J. AND RICONSCENTE, M. 2006. Evidence-centered assessment design. *Handbook of test development*, 61–90.

PUMPREPAIR 2013. *Pump Repair Sample Task*. `http://nces.ed.gov/nationsreportcard/tel/wells_item.aspx`.

SAO PEDRO, M. A., DE BAKER, R. S., GOBERT, J. D., MONTALVO, O., AND NAKAMA, A. 2013. Leveraging machine-learned detectors of systematic inquiry behavior to estimate and predict transfer of inquiry skill. *User Modeling and User-Adapted Interaction 23,* 1, 1–39.

SCHMIT, M. J. AND RYAN, A. M. 1992. Test-taking dispositions: A missing link? *Journal of Applied Psychology 77,* 5, 629.

SUNDRE, D. L. AND WISE, S. L. 2003. Motivation filtering: An exploration of the impact of low examinee motivation on the psychometric quality of tests. In *annual meeting of the National Council on Measurement in Education, Chicago, IL*.

TEL 2013. *Technology and Engineering Literacy Assessments*. `https://nces.ed.gov/nationsreportcard/tel/`.

USMLE 2014. *United States Medical Licensure Examinations*. `http://www.usmle.org/pdfs/step-3/2014content_step3.pdf/`.

VINH, N. X., EPPS, J., AND BAILEY, J. 2009. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 1073–1080. ACM.