+[IEA Research Report]

Students' Behavioral Patterns in a Computational Thinking Interactive Task: Using Process Data in ICILS 2018 with Sequence Mining

Qiwei He and Eugenio Gonzalez

Educational Testing Service

October 2023

Abstract

Organized by the International Association for the Evaluation of Educational Achievement (IEA), the International Computer and Information Literacy Study (ICILS) in 2018 extends the evaluation of students' computer and information literacy (CIL) skills and introduces a novel assessment of students' computational thinking (CT) skills. CT skills are defined as the ability to recognize, analyze, and describe real-world problems so their solutions can be operationalized within programming tasks. Eligible process data recorded during students programming a CT problem-solving task could provide a new angle to understand how students learn, interact, and adapt their CT skills to solve digital tasks in the programming environment. The aim of this study is two-fold: given limited availability of process data in fine-grained level, we focus on testlet level, namely, to extract behavioral patterns from the whole CT unit. First, to cluster the behavioral patterns into meaningful groups, thus extract representative time allocation patterns through the items within one CT unit, and second to evaluate the CT skills by each latent group to identify the most optimal pattern across countries.

The current study focuses on a total of 31,344 students of nine countries and regions who were assigned to the CT modules, "Farm Drone," in ICILS 2018. Two sub-studies were conducted in this project: (1) We conducted a cluster analysis on timing and process-related variables, to group students with homogeneous patterns and map with students' CT and CIL latent scores and background variables across countries. (2) We focus on command efficiency, correctness, and time allocation patterns from process data to calculate the sequence distance and extract representative patterns from the way that students solve the coding tasks throughout the unit. This sub-study focuses on better understanding how students with missing responses allocated their time and pinpoints potential reasons why they missed certain items.

1. Background

Computational thinking and adaptive problem-solving skills in digital competence tasks consist of the abilities of individuals to use computers to collect, manage, produce, and exchange information as well as formulate solutions to problems. These have been recognized across countries as among the most important skills in the 21st century. The International Computer and Information Literacy Study (ICILS) in 2018 organized by the International Association for the Evaluation of Educational Achievement (IEA) extends the evaluation of students' computer and information literacy (CIL) skills and introduces a novel assessment of students' computational thinking (CT) skills, defined as the ability to recognize, analyze, and describe real-world problems so their solutions can be operationalized within programming tasks (Fraillon et al., 2019).

Eligible process data recorded during students programming process could provide a new angle to understand how students learn, interact, and adapt their CT skills to solve digital tasks in the programming environment. Specifically, we aim to: (1) identify and characterize distinct and general profiles of students' CT strategies and test-taking behaviors by proficiency scores, (2) examine whether students' CT strategies and test-taking behaviors are similar by countries, and (3) assess whether behavioral patterns of solving the CT tasks are related to students' CIL and CT proficiency.

1.1 Computational Thinking and ICILS

Changes in society, environment, and technology are shifting the emphasis of education away from equipping students with routine skills to empowering them to confront and overcome complex challenges in a digital world. As a pioneering assessment, ICILS established an agreed definition and explication of CIL as a student learning outcome in its first cycle in 2013 and developed an innovative assessment of CT to measure students' capacities to plan and operationalize computer-based solutions to real-world problems in the new cycle in 2018. This extension laid a foundation on structuring a framework for measuring and reporting achievement in CT and baseline measures against which CT can be monitored over time (Fraillon et al., 2020).

CT, also the focus in this study, indicates "the type of thinking used when programming on a computer or digital device" (Fraillon et al., 2020, p.18). While concepts associated with CT have been recognized since the advent of computing as a field of study in the 1940s (Denning, 2017), the role of CT in curricula has not been paid much attention until the end of the 20th century when personal computers were introduced to schools. A resurgence of interest in the teaching and learning of CT emerged recently as both a foundation for the effective use of digital technologies and as a transferable set of problem-solving skills (Fraillon et al., 2020).

Two strands were highlighted in CT in ICILS 2018: conceptualizing problems (through algorithmic or systems thinking) and operationalizing solutions (creating, implementing, and evaluating computer-based solutions to problems), corresponding to two CT test modules respectively (Fraillon et al., 2020). Among them, the CT module for the operationalizing

solutions strand was innovatively designed in a visual coding environment to capture students' programming process to facilitate examining their abilities in developing algorithms, programs, and interfaces.

1.2 Theoretical background

The contextual framework for ICILS 2018 CIL/CT learning outcomes can be used as a theoretical model to locate CT, student learning process, and background characteristics (Heldt et al., 2020). Students acquire competencies in CT through a variety of activities and experiences at the different levels of their education and through different processes in school and out of school (Fraillon et al., 2019). The present model assumes that the features at the antecedent level (input) directly affect the learning processes. These learning processes, in turn, are assumed to correlate with students' CIL/CT as the learning outcomes (output), and thus, have an impact on competences. It is important to note that there is a reciprocal association between learning processes and learning outcomes while a unidirectional influence between antecedents and processes (Figure 1). As the CT measurement framework stresses, both antecedents and processes need to be considered when explaining variation in CIL/CT learning outcomes. Whereas antecedent factors shape and constrain the development of CIL/CT, process factors can be influenced by the level of (existing) CIL/CT learning. For example, the level and scope of classroom exercises using ICT generally depend on the existing CIL-related proficiency of the students (Fraillon et al., 2019).

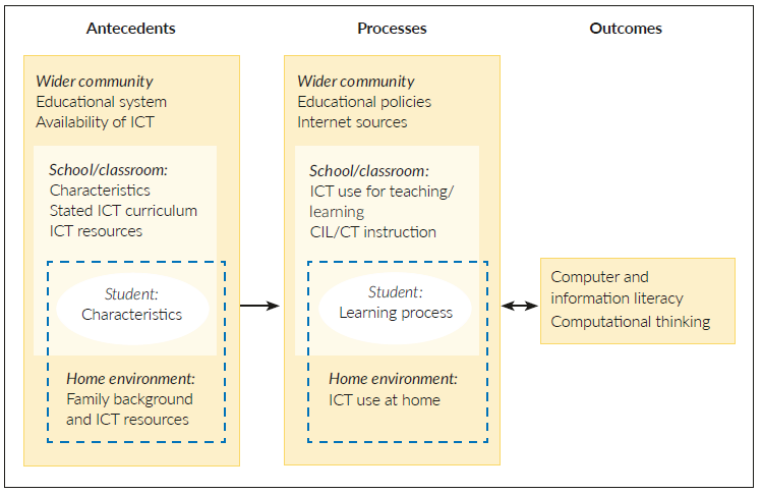

Figure 1. *Contextual framework for ICILS 2018 CIL/CT (Source: Fraillon et al., 2019)*

Process data captured from the CT module could enrich the information on the antecedent's level by better describing students' problem-solving strategies and test-taking behaviors (He et al., 2021; Sahin & Colvin, 2020). Specific behavioral patterns could also have an impact on students' learning process and eventually influence the outcome of CIL/CT. The addition of information extracted from process data is expected to strengthen the predictive power of antecedents and processes, though this function is not explicitly considered in the ICILS model (Heldt et al., 2020).

Heldt et al. (2020) presented an input-process-output model regarding the role of process data during task completion, which provides theoretical support in our proposed study. In this framework model (Figure 2), it is assumed that the completion process during testing (process) is

impacted by person-level characteristics and task-level characteristics (input), which has a direct influence on the result of task completion (output).

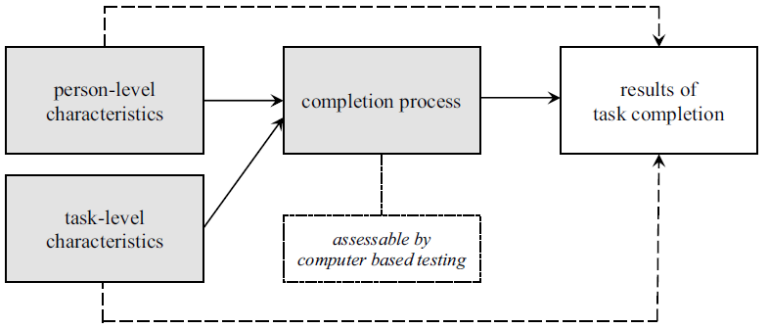

Figure 2. *Theoretical function of process data in completion process (Source: Heldt et al., 2020)*

Finer-grained process data with a record for each step provides a potential solution to facilitate the assessments of CT. The technology/computer-based testing in the context of large-scale assessment enables the determination of process data such as action/coding sequence and response times, which can describe individual behavioral differences during the process of task completion and thereby task success (Goldhammer et al., 2017; He et al., 2021; Tang et al., 2020; Ulitzsch et al., 2021). These sequence-derived latent features can predict the final responses of test-takers, as well as performance on other items and various cognitive traits.

1.3 Sequence-based methods in process data

The programming process that is inferred from the observed patterns of test responses represents a dynamic, unfolding mechanism (Launeanu & Hubley, 2017). The information extracted from process data is particularly valuable when examining such interactive problem-solving tasks (He et al., 2021). The new data source provides exciting opportunities to deeper understand *what* and *how* students adapt CT strategies in solving digital tasks beyond merely correctness/incorrectness, with the aim of pinpointing the underlying reasons for students' success and failure, extracting students' time management strategies through the module, and identifying the debugging phases that might be a struggle for students, to name just a few.

Sequence-based features in process data analysis are primarily grouped into two categories: mini-sequences that are disassembled from a long sequence, and measures of similarity computed by distances of pairs of full sequences. The mini-sequences are usually represented by n-grams, that is, a contiguous sequence of n items from a given sequence such as clickstream, text, or speech (He & von Davier, 2016). Features derived from clickstreams comprise: a) generic features commonly used in sequence mining or natural language processing (e.g., n-grams as in He & von Davier, 2015, 2016; Ulitzsch et al., 2022); b) task-specific features, created based on subject-matter knowledge on behavioral patterns to be expected on the task (Chen et al., 2019; Salles et al., 2020); or c) a combination of the two (Han et al., 2019; Liao et al., 2019; Qiao & Jiao, 2018). These features are then fed to classifiers or prediction models or analyzed using sequence mining techniques to identify features that best distinguish correct from incorrect clickstreams (Ulitzsch et al., 2022).

Sequence distance functions are designed to measure sequence (dis)similarities in sequence mining. A common approach for detecting patterns in action sequences consists of converting the

information contained in action sequences into distance measures (Dong & Pei, 2007). In the context of problem-solving processes, sequence distance measures can be defined to describe how action sequences differ either from each other (Tang et al., 2020; Ulitzsch et al., 2021; Gao et al., 2022) or with respect to pre-defined sequences (Hao et al., 2015; He, Borgonovi, & Paccagnella, 2019, 2021).

Character alignment-based distance functions are broadly used in sequence proximity metrics. These algorithms can be local window-based or whole sequence based; they can also be edit distances or more general pairwise similarity score-based distance (Tang et al., 2020; He et al., 2021; Dong & Pei, 2007). For instance, the edit distance function, also called the Levenshtein distance (Levenshtein, 1965, 1966), between two sequences $S1$ and $S2$ defines the minimum number of edit operations (i.e., deletion, insertion, and substitution) that are needed to transform $S1$ into $S2$ (Jurafsky & Martin, 2009). Hao et al. (2016) applied the edit distance method to compute the similarity between action sequences to identify the typical strategies by correct and incorrect groups to solve a scenario-based digital task. The Hamming distance between two sequences is limited to cases where the two sequences have identical lengths and is defined as the number of positions where the two sequences are different (Hamming, 1950). Two sequence-distance-based similarity computation methods, i.e., longest common subsequence (LCS) and dynamic timing warping (DTW) are highlighted below.

The LCS (Hirschberg, 1975, 1977) identifies the longest subsequence that is common to two strings. The length of the LCS is defined as the degree of closeness between the two strings. Sukkarieh et al. (2012) used the LCS to cluster students' response sequences with high similarity and provide automatic scoring in multiple language environments. He et al. (2021) employed the LCS to compute the similarity between the predefined sequence and individual observed sequence to explore how far the respondent's solution was away from the optimal path (i.e., the action sequence of correct solution with the minimum number of actions). Ulitzsch et al. (2021) extended the LCS application by integrating the time intervals between actions to better understand respondents' unusual test-taking behaviors—such as long-time pause and speedy skipping—to assist in pinpointing the potential reason for their success and failure in a digital task. Gao et al. (2022) used neighborhood density (Gabadinho et al., 2011) as a representativeness criterion for sorting candidate representative sequences and chose the optimal matching between sequences of spells (Omspell) algorithm (Studer & Ritschard, 2014) to measure similarities between sequences of navigation pages, thus resulted in four homogenous groups of navigation patterns.

The DTW method (Sakoe & Chiba, 1978) is an alternative algorithm to compute the similarity between time-series sequences via dynamic programming. It was first developed by the speech recognition community to handle the matching of non-linearly expanded or contracted signals. The algorithm features in finding the optimal path through a matrix of points representing possible time alignments between the signals. In the context of navigation pattern exploration, each visited page could be recorded in a sequence and the time spent on each visited page could also be recorded into a sequence (He et al., 2023).

1.4 The present study

The aim of this study is two-fold: given limited availability of process data at a fine-grained level, we focus on the testlet level, namely, to extract behavioral patterns from the whole CT unit. The first aim is to cluster the behavioral patterns into meaningful groups, and thus extract representative time allocation patterns through the nine items within one unit, and the

second aim is to evaluate the CT skills by each latent group to identify the most optimal pattern across countries.

To achieve this goal, we conduct two sub studies: (1) We conducted a cluster analysis on timing and process-related variables, to group students with homogeneous patterns and map with students' CT and CIL latent scores and background variables across countries. (2) We focus on command efficiency and time allocation pattern from process data to calculate the sequence distance and extract representative patterns from how the students solve the coding tasks throughout the whole unit. This sub-study focuses on better understanding how students with missing responses allocate their time and pinpoints potential reasons for their missing certain items. In this study, we stress the grouping of students by their CT and CIL proficiency levels, rather than background variables (e.g., gender, socioeconomic status, and other questions related to programming coding), which will be pursued under a separate study in the near future.

The report is structured as follows: In Section 2, a summary of study sample, instrument and process-related variables is presented. The first sub-study using cluster analysis on timing and process-related variables to group test-takers homogenous' behavioral patterns is presented in Section 3, which is followed by the second sub-study in Section 4, where a sequence mining analysis with DTW method is used to extract nonresponse patterns and time allocation especially from test-takers who showed missing responses in the computational thinking module. This report concludes with a discussion and further thoughts about the use of process data in future ICILS assessments.

2 Data and Materials

2.1 Sample

The current study focuses on a total of 31,344 students who were assigned the computational thinking module, "Farm Drone," in ICILS 2018. The students were from nine countries and regions,[1] consisting of Denmark (DNK), Finland (FIN), France (FRA), Germany (DEU), Republic of Korea (KOR), Luxembourg (LUX), North Rhine-Westphalia (Germany) (DNW), Portugal (PRT), and the United States (USA). The sample consists of 48.8% girls and 51.2% boys. The average age is 14.34 years old (S.D. = 0.62). Among the 31,344 students, over 80% students reported they had used a computer, smartphone or tablet for at least 3 years, and 43.9% of students reported having ICT studies in the current school year. More than two-thirds of students had internet access at home. It is also noted that the inequality of technical equipment and access cannot be ignored. Of those surveyed, 8.1%, 4.8%, and 15.2% of students reported that they had either never or used for less than one year a computer, smartphone, or tablet, respectively. Most of those who lacked technical resources came from low-income families, whose national index of socioeconomic background is on average 0.8 lower than the group who had used electronics for longer.

[1] The whole sample size from all nine countries participating in the optional CT module for ICILS 2018 were utilized in this study.

Table 1 describes students' profiles by countries in ICILS 2018. The average score of all nine countries and regions located at Level 2 in CIL and middle region in CT.[2] Denmark and the Republic of Korea ranked as the top performers in CIL and CT, respectively, by their average scores of 553.7 and 532.4. Luxembourg ranked the lowest among the participating countries and regions in both CIL and CT, with average scores of 482.7 and 461.2, respectively.

Table 1. Description of students' profiles by country and region in ICILS 2018

	N	Gender (Female)	CIL		CT	
			Mean	S.D.	Mean	S.D.
DEU	3,596	48.90%	520.4	75.3	489.3	100.0
DNK	2,384	49.50%	553.7	64.8	528.3	83.9
DNW	1,961	49.30%	516.7	75.6	487.9	96.5
FIN	2,485	49.20%	529.2	79.9	506.9	95.3
FRA	2,923	49.80%	497.3	79.2	501.5	91.0
KOR	2,852	47.90%	539.9	93.5	532.4	109.0
LUX	5,318	47.40%	482.7	82.8	461.2	106.1
PRT	3,170	48.60%	512.9	69.8	481.5	78.7
USA	6,655	49.40%	518.4	80.5	497.0	106.9
TOTAL	31,344	48.80%	515.4	81.5	494.7	101.2

Note. CIL indicates computer and information literacy skills; CT indicates computational thinking skills. The average CIL and CT scores were computed based on the first plausible value of CIL and CT respectively.

2.2 Instrument

The current study focuses on one CT module "Farm Drone." In the Farm Drone module, students worked within a simple visual coding environment, that is, students could drag and drop code blocks each of which performed a specified function, to create, test, and debug code that controls the actions of a drone used in a farming context. The difficulties of the tasks relate to the code functions that were available and the complexity of the sequence of actions required by the drone to complete the task. Students' responses were captured by the assessment system and later scored based on following two characteristics:

The "correctness" with which the drone performs the actions specified in the task. This includes both the degree to which the drone performs required actions and the presence or absence of any unrequired actions. And the "efficiency" of the code, which was measured by comparing the number of code blocks used in the solution with the minimum number required to implement a fully correct solution, namely, with longer code sequences corresponding to lower

[2] In ICILS 2018, the CIL proficiency was defined by five levels: Below Level 1 (< 407 scale points), Level 1 (407 to 491 scale points), Level 2 (492 to 576 scale points), Level 3 (577 to 661 scale points), and Level 4 (> 661 scale points). The CT proficiency was defined by three regions: Below Region (<459 scale points), Middle Region (459 to 589 scale points), and Upper Region (> 589 scale points).

scores. Each Farm Drone task included an instruction for students to use as few code blocks as possible. Ultimately, each coding task received a single score derived by combining the correctness and efficiency scores. For most tasks, the efficiency score was used to moderate the score attributed to completely correct responses. Full details of the scoring for each Farm Drone coding task are provided in the ICILS 2018 technical report (Fraillon et al., 2020).

Figure 3 exhibits a screen shot of Task 6 in the Farm Drone module. The interface design for the Example CT module was divided into two functional spaces. In the Farm Drone module students could return to previously completed tasks by clicking on the green task box corresponding to the ordinal position of the task. Students could also use a flag toggle to mark tasks that they wanted to go back to if they had sufficient time to review and improve. The programming code could be dragged and dropped from the code blocks into the workspace, or removed back to the drone simulation space. The program codes could be embedded into the "repeat" ("for" loop function) and "if" clause function. Students were required to finish all nine items in the module within 25 minutes. The remaining time was displayed in the time bar to remind students checking their pace while solving the problems.

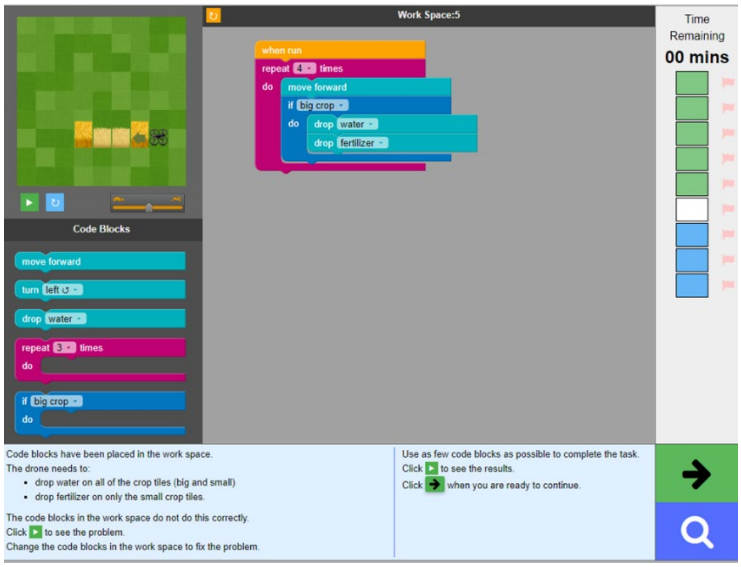

Figure 3. *A screenshot of Task 6 in Farm Drone module in ICILS 2018 (Source: ICILS 2018 Technical Report)*

As shown in Table 2, the Farm Drone module consists of four task types: warm up, creation, debugging and configuration. The warm-up item in the first task provides a trial opportunity for students to learn how to control the farm drone and drag and drop the programming code into the workplace. In the creation tasks, students are required to build up the programming codes in the workplace following corresponding instructions. Conversely, in the debugging tasks, a set of programming codes have been pre-populated in the workplace, and students are required to revise the codes to satisfy the conditions in the task. The last task is the most complex, students must configure the pre-populated commands and create new codes to solve the task. The target layout (the number of rows covered by the farm drone), and the number of targets (single or multiple) also impact on the difficulty and complexity of the CT

task. With the gradually increasing complexity throughout the Farm Drone module, the average correctness decreased, and the number of missing responses also rose significantly.

Table 2. Task type, layout, process information and completion status in Farm Drone module

Task No.	Task type	Target layout	Number of Targets	Average Time (Seconds)	Average Number Command	Average Reset	Average Correctness	Average Nonresponse Percentage
Task 1	Warm up	N/A	N/A	76.9	3.5	2.0	95.1%	1.8%
Task 2	Creation	single	1	40.9	4.0	0.9	88.3%	3.3%
Task 3	Debugging	single row	4	119.0	8.1	2.6	84.8%	4.0%
Task 4	Creation	single row	4	99.4	10.0	1.5	85.6%	3.0%
Task 5	Creation	double row	8	236.1	16.6	4.1	76.7%	3.9%
Task 6	Debugging	single row	6	127.9	8.9	2.5	62.3%	17.8%
Task 7	Creation	double row	12	195.6	16.4	3.6	50.9%	14.4%
Task 8	Debugging	quadruple row	16	120.4	16.7	3.9	26.7%	40.8%
Task 9	Configuration	quadruple row	16	78.9	NA	3.3	35.5%	49.0%

2.3 Process variables

In this study, we focused on using 39 process variables by six types to describe students' test-taking behaviors and strategies, including time spent on each item, remaining time in the module, number of commands, number of clicks of the reset button, algorithms chosen in each task, and whether irrelevant targets were covered. Table 3 provides a detailed description of each process variable type.

The 5th to 8th column in Table 2 presents the statistics of the process variables. On average, students spent a significant amount of time on the first task (76.9 seconds on average) to adapt themselves to the new environment and learn the different tools (e.g., trying multiple times to drag and drop the programming codes and clicking the reset button to get back to the default status). After quickly learning how to use the tools, students seemed more confident in the second task as there were no big changes in the environment. The third task moderately changed the requirement from creation to debugging, which resulted in the students taking more time, as they had to figure out the new instructions and reset more frequently to check the farm drone settings. On average, it appeared that students spent the longest time in solving Task 5. The students used the reset function the highest number of times in this task as they made repeat attempts, which resulted in one of the longest commands in the workspace. A probable reason was a significant enhancement of complexity in Task 5, i.e., to control the targets from a single row to double rows. Spending more time on this task might have led to insufficient time to solve remaining tasks in the module, which resulted in a significantly higher missing rate in the following tasks (from 3.9% nonresponse rate in Task 5 to 17.8% nonresponse rate in Task 6). We also noticed that students obviously used the reset function more frequently (on average nearly 4 times) when solving Task 7 and Task 8. The average time spent in Task 9 significantly

decreased as students might rush up to the end of the module. Over 40% of students skipped the last two items, and the missing rate rose to nearly 50% in the last task.

Table 3. Description of Process Variables in Farm Drone Module

Variables	Number of variables	Tasks Involved	Description
Algorithms	6	3,4,5,6,7,8	The programming coding strategy that is chosen by student, either by straightforward move codes or by nested codes. There are 6 levels predefined in this variable. Only one algorithm strategy could be labeled per student per item. Level 1: move with turn/drop Level 2: nested move in a repeat Level 3: nested move with drop in a repeat Level 4: nested if do in a repeat Level 5: nested repeat in a repeat Level 6: nested if do in a repeat nested in another repeat
Commands	8	1,2,3,4,5,6,7,8	The number of command lines recorded in the item (the final commands shown on screen only). The adding or removing of commands during the problem-solving process are not counted.
Irrelevant Targets	6	3,4,5,6,7,8	A binary indicator to record "one or more irrelevant target" (label 0) and "No irrelevant targets" (label 1).
Remaining time	1		The time remaining in the Farm Drone module after submitting responses to all 9 items. A total 25 minutes is set for the whole module.
Reset	9	1,2,3,4,5,6,7,8,9	Number of clicks on reset button to restart the Farm Drone simulation. (Only the reset button in the simulation space is counted).
Task Time	9	1,2,3,4,5,6,7,8,9	Total time spent on an individual task. Students are allowed to go back and forth to different tasks within one module. The task time is the sum of time spent in every visit.

Process patterns (e.g., length of commands, reset function, and time on task) throughout the module were further compared across countries and regions. As Figure 4 displays, the nine countries and regions follow similar patterns in length of commands and time spent on task. Students from Denmark and Portugal were found to take relatively more time than their peers, while students from Luxembourg were found, on average, to be much quicker in solving every item. There was a robust difference in use of the reset function between countries in the second half of the Farm Drone module. Students from the Republic of Korea and France used the reset function much more frequently than their peers in Task 5 to Task 9.

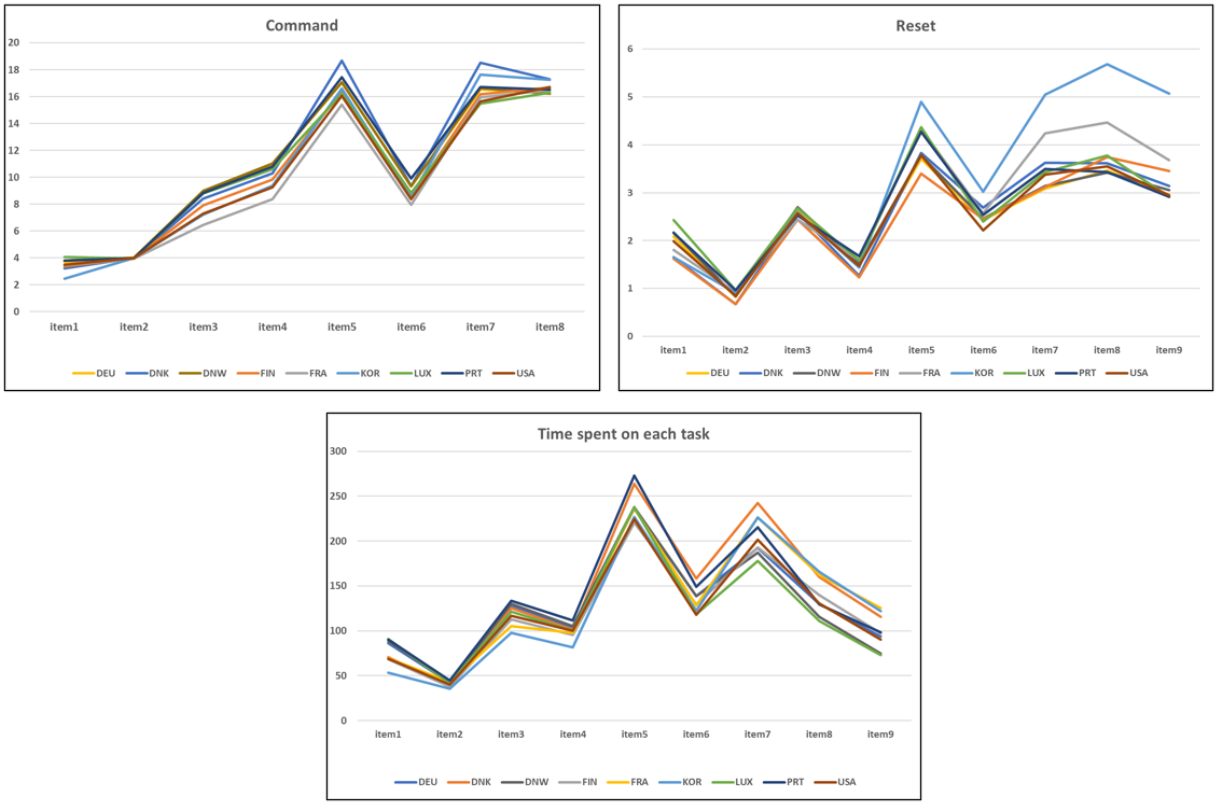

Figure 4. *Sequential process patterns in command length, reset function, and time spent on tasks throughout the Farm Drone Module by country and region*

2.4 Missing responses

The high missing response rate aroused our attention to make further analysis. Among the 31,344 students, 11,468 (36.6%) students gave full responses to all nine tasks, while 19,876 (63.4%) students had at least one missing response in the nine tasks throughout the Farm Drone module.[3] Figure 5 shows the distribution of missing responses across the nine tasks by country and region. As mentioned earlier, the missing response rate significantly increased after Task 5. Students of Luxembourg showed higher missing response rates than their peers across all items, and more obviously in the second half of the module from Task 5 to Task 9. Frequently skipping items tremendously reduces the response time, which might be an interpretation for why students from Luxembourg were found to be the fastest solvers among all the participating countries. The missing response rate rose steeply for the Republic of Korea group from Task 7 to Task 9. As noted earlier, this group was found using the reset function more frequently than their peers. It could be interpreted that students from the Republic of Korea spent time trying to solve the

[3] The order of CT modules was not taken into consideration in the present study. In the ICILS 2018, two CT modules were used. The location of Farm Drone module as the first or the second module in the CT assessment might impact on the response time and missing rate.

problem, reset the farm drone multiple times, but might still have been unable to submit a response.

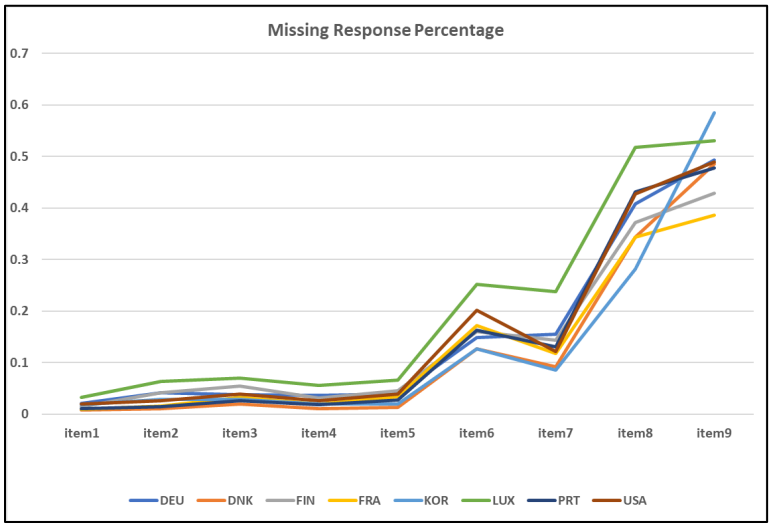

Figure 5. *Missing response rate across nine tasks in Farm Drone module by country and region*

3. Sub-study 1: Identify homogeneous behavioral patterns with K-prototype clustering analysis

In this study, we decided to separate the exploration of students' behavioral patterns by complete and incomplete (with responses) data for two reasons. First, the missing responses might violate the generalization of behavioral patterns that we could draw from the complete response sets across the unit. Students who could give the whole set of answers would choose different strategies to solve the coding tasks across the unit and would have special time allocation strategies, which could be completely different from the sample with missing responses. Second, from a technical perspective, clustering analysis does not allow for the use of missing values in the attributes. In this situation, we could estimate the values of missing responses. However, this solution presents some uncertainty and required further treatment of the prediction errors, which could significantly impact the clustering results. To extract patterns and group the homogeneous behavioral patterns and strategy of solving CT tasks, clustering analysis based on unsupervised machine learning is an appropriate approach.

The first sub-study focuses on a total of 11,468 students who gave full responses to the nine tasks throughout the Farm Drone module, that is, students without any missing responses were included in this cluster analysis. The aim was to illustrate the behavioral patterns of students who were relatively highly engaged in the CT assessment, without any potential confusion that might arise from missing responses (either because of low engagement or giving up because of high difficulty). For the remaining two-thirds of students who had at least one missing response, we conducted a separate study (sub-study 2) to identify the nonresponse patterns with the sequence mining method in Section 4.

Specifically, two research questions are to be pursued in sub-study 1: (1) what homogeneous behavioral patterns could be extracted from the CT tasks, and (2) whether the behavioral patterns could be mapped onto students' CT and CIL proficiency scores.

3.1 K-prototype clustering algorithm

In the present study, we employed the K-Prototype clustering method on 39 aggregated process variables (see Table 3) to group students by homogeneous behavioral patterns. Note that within the 39 process variables, there are both continuous variables (i.e., response time, remaining time, reset clicks, length of commands) and categorical variables (i.e., algorithms and irrelevant targets). The commonly used clustering algorithms such as K-means (MacQueen, 1967; Anderberg, 1973) may not be suitable. The problem happens when the cost function in K-Means is calculated using the Euclidian distance that is only suitable for numerical data. The K-Modes algorithm (Huang, 1997) uses a simple matching dissimilarity measure to deal with merely categorical objects, replaces the means of clusters with modes, and uses a frequency-based method to update modes in the clustering process to minimize the clustering cost function. To deal with the mixed data types, the K-Prototypes algorithm proposed by Huang (1998), through the definition of a combined dissimilarity measure, further integrates the K-means and K-modes algorithms to allow for clustering objects described by mixed numeric and categorical attributes.

A principal factor analysis (PCA) was first conducted on the continuous variables to reduce the dimensions for clustering. Afterwards, the principal components with categorical variables were input in the K-Prototype clustering algorithm to identify the homogeneous patterns. By checking the within cluster sum of squares by setting $k = 2$ to $k = 10$, the optimal number of clusters was identified as $k = 4$. The clustering results by four homogeneous patterns are reported hereafter.

3.2 Homogeneous patters from clustering

Figure 6 presents the four extracted clusters mapped on the first two principal components. It is noted that the first component and second component explained over 25% variance across groups.[4] Cluster 1 and Cluster 4 showed a big overlap with the first two components, suggesting other components or categorical variables might play an additional role in distinguishing these two clusters. Cluster 2 occupied the biggest proportion (41%) in the sample, followed by Cluster 1 (29.1%), and Cluster 3 (26%). Cluster 4 had the smallest sample size, occupying 3.9% of the sample.

The features of each cluster are further presented in Figure 7, where the length of commands, frequency of using the reset function, and time spent on tasks across nine items were exhibited in sequential lines. In the plot describing the length of command, Cluster 1 (red line) showed significantly longer commands in Task 3 to Task 6 compared with their peers. Cluster 2

[4] From the visualization perspective, we showed the first two principal components only. The clustering analysis includes more principal components that have lambda>=0.1, to increase the accuracy of clustering.

(green line) and Cluster 3 (blue line) were found to have a very similar pattern, while Cluster 4 (purple line) used longer commands in a relatively easy item (i.e., Task 3) but much shorter commands in tasks with high difficulty and complexity (e.g., Task 7).

In the plot regarding the frequency of using the reset function, Cluster 4 was found to seldom use reset function throughout the whole Farm Drone module, while Cluster 3 was more likely to frequently click the reset button than their peers. Cluster 1 showed an active use of reset function in Task 5 and maintained a high rate of use of the reset function till the end of the assessment module. However, Cluster 2 only marginally used the reset function from the very beginning of the module until Task 7 and significantly enhanced their use in the last two items.

Finally, considering the time spent on each task, Cluster 1 showed a peak use of time on Task 5 while Cluster 3 showed a peak use of time on Task 7. Cluster 4 were shown to be fast problem-solvers across the nine items. The time that they spent on Task 5 was only half of Cluster 1. Cluster 2 stood out in the last three items, where they significantly increased their problem-solving time while the other clusters all showed a decreasing tendency.

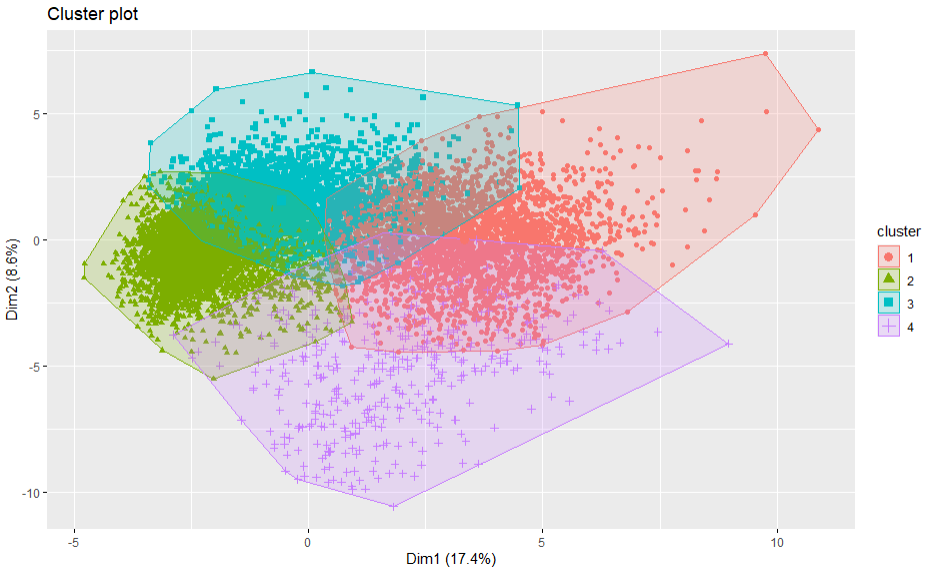

Figure 6. *Extracted four clusters mapping on the first two components*

To clarify the features by different groups, we summarized the profiles of each cluster as below:

- Cluster 1 input longer commands from Task 3 to Task 6, used the reset function more frequently than their peers, and spent the longest time on Task 5.
- Cluster 2 followed a similar pattern to Cluster 3 using commands of a moderate length to solve the tasks. Students in this group did not use the reset function frequently from the beginning of the module but significantly increased their use

of the reset function in the last two tasks. In addition, Cluster 2 spent significantly more time on the last three tasks.

- Cluster 3 were more likely to use the reset function than their peers. This group clicked the reset button the most times in Task 6 and spent the longest time in solving Task 7.
- Cluster 4 input relatively short commands, rarely used the reset function, and solved each item very fast.

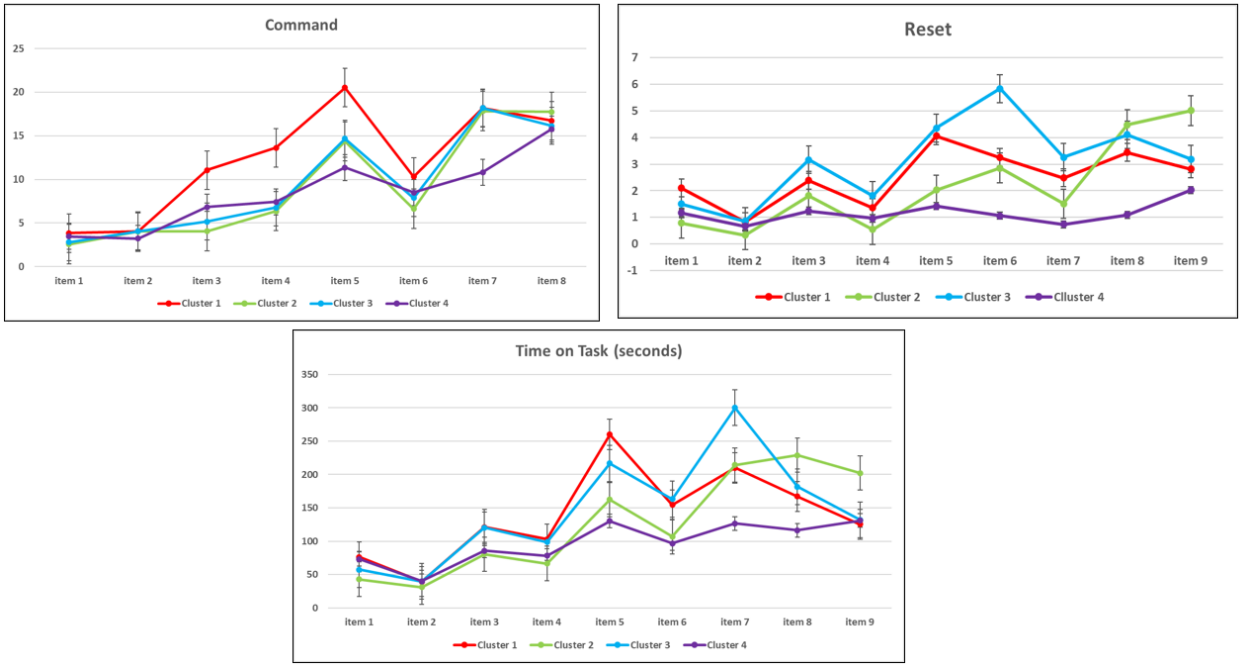

Figure 7. *Behavioral features by four clusters across nine tasks in the Farm Drone module*

3.3 Mapping behavioral patterns with CT and CIL proficiency

We then mapped the four behavioral pattern groups into the proficiency score in CT and CIL, respectively. Similar results were found in these two dependent variables. Cluster 4 showed the lowest proficiency score (399.2 in CT and 457.9 in CIL) in both CT and CIL proficiency scores, followed by Cluster 1 (505.0 in CT and 527.4 in CIL). Cluster 2 showed the highest scores (607.6 in CT and 584.7 in CIL) among the four clusters, closely followed by Cluster 3 (569.0 in CT and 565.5 in CIL).

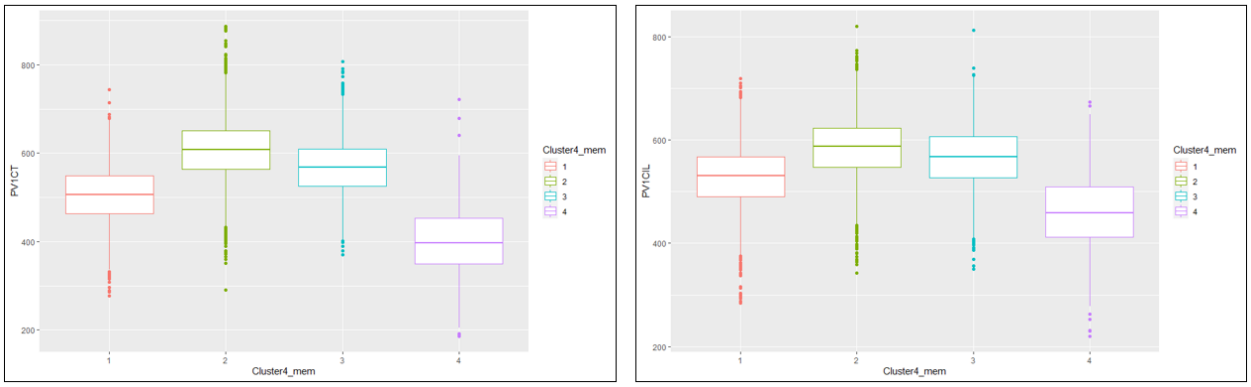

Figure 8. *Behavioral pattern clusters mapping on CT and CIL proficiency scores*

To link with the features by each cluster, the result suggested that commands which were too long (Cluster 1) or too short (Cluster 4) were not helpful in enhancing the proficiency score. Use of the reset function was a good sign that students were engaged in the task. More frequent use of reset function (i.e., Cluster 3) did not necessarily lead to better CT or CIL scores but was a very helpful strategy in exploring difficult and complex items. For example, Cluster 2 showed high use of the reset function in the last two items.

Time allocation also had an impact on CT and CIL proficiency scores. Solving the CT tasks consistently in a fast manner (e.g., Cluster 4) might not help achieve a high score, because some items did need to be checked carefully, especially the procedure of debugging and creating new codes. Optimal time allocation and reserving time for items at the end (e.g., as Cluster 2 did) would have provided more flexibility in handling complex items. Spending too much time on items in the middle of the module (e.g., Cluster 1 on item 5 and Cluster 3 on item 7) might have put the students in a situation where they had to rush. Though the students might not have skipped the items at the end, they probably did not have sufficient time to carefully go over the details.

Figure 9 displays the proportion of each cluster by country and region. The smaller proportion of patterns shown in Cluster 4 and bigger proportion of behavioral patterns shown in Cluster 2 and Cluster 3 would be helpful in enhancing the general performance of each country. For example, as shown in Table 1, the Republic of Korea and Denmark ranked the highest in CT and CIL, respectively. The proportion of Cluster 4 in these two countries was as low as 1.8% and 1.4%, respectively, which were the lowest among the nine countries and regions.

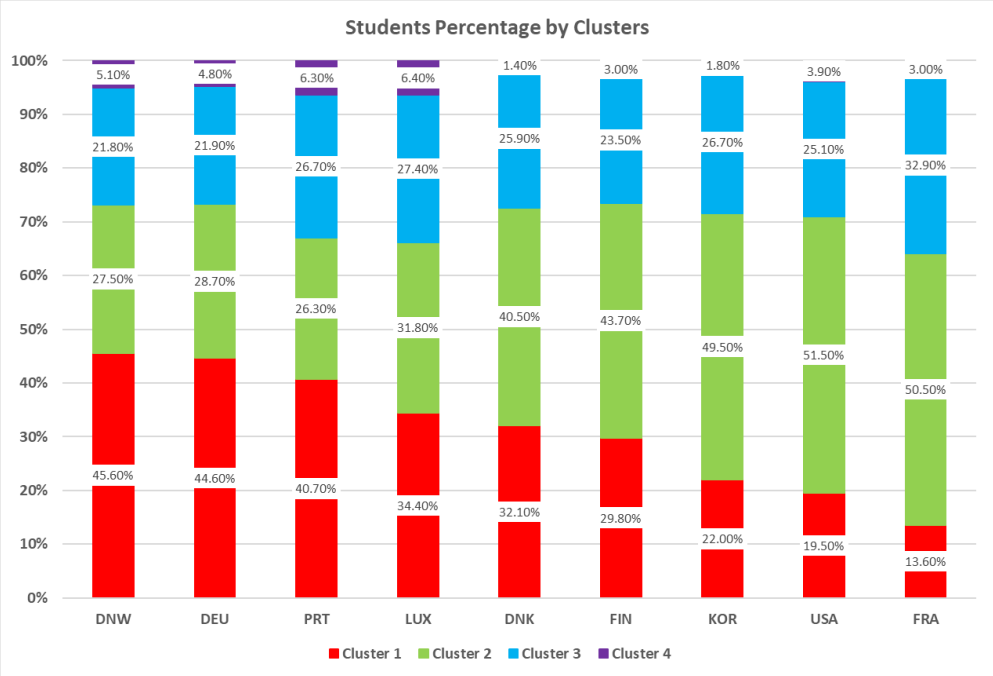

Figure 9. *Proportion of behavioral pattern clusters by country and region*

4. Sub-study 2: Sequence identification for time allocation and command efficiency

The second sub-study focuses on a total of 19,876 students who had at least one missing response throughout the Farm Drone module with the aim of extracting the nonresponse sequential pattern and pinpointing the potential reason for the missing responses. Two research questions were investigated in this sub-study: (1) are there any nonresponse patterns that could be extracted in the CT tasks, and (2) how students allocated their time in different nonresponse patterns.

Specifically, we first input the combined score of efficiency and correctness (i.e., 0, 1, 2, 3, 4) per item per student into a sequence, that is, each student has a sequential combined score across the nine tasks. The missing response was recorded as 9. For example, one student's score sequence could be (1, 2, 3, 0, 9, 0, 1, 3, 9), in which two missing responses were recorded in item 5 and item 9 and the other numbers indicate the score on the corresponding task. We computed the pairwise sequence similarity by every two students' sequence. Based on the huge, derived distance matrix, we could extract the representative nonresponse sequential patterns. This information will help us identify where the missing response is located (i.e., at the beginning of the module, in the middle, or at the end), whether there is a particular pattern (e.g., items that could show simultaneous missing responses), and thus give us a better understanding of the potential reasons for the missing response, which will help to enhance the test design in the future. We employed the dynamic time warping method to compute the sequence similarity on the pairwise distance and conduct sequence clustering.

4.1 DTW method

The DTW method is one of the similarity distance measures which can be used to assess how similar two sequences are, especially when the data entails a time-series format. It has been widely applied to problems in economic and sales forecasting (e.g., Arya et al., 2021; Chang et al., 2008), speech recognition (e.g., Permanasari et al., 2019; Amin & Mahmood, 2008), and music rhythm identification (e.g., Ren et al., 2016; Guo & Siegelmann, 2004). Unlike data types in traditional databases where the similarity of distance definition is straightforward, the distance between time series needs to be carefully defined in order to reflect the underlying proximity of these specific data, which is usually based on shapes and patterns (Kurbalija et al., 2011). For instance, in stock market analysis, the stock curves of two companies may follow similar patterns but show peak occurrences at different time points. Following traditional approaches, for instance, using the Euclidean distance, to calculate the similarity measure between the two stock curves, the distance would yield very large distances between the two sequences, because it ignores that the shape of the two curves is very similar but located at a different pace. Analogously, the sequential data resulting from performance of the whole unit (nine items in total) echo the same needs. When we put the efficiency score sequence along a time axis, one student's performance path could be similar to another, but with a different time pause at each item. Finding the optimal warping path between the two sequences can help to reflect the appropriate similarity measures between the sequences.

Following He et al. (2023), given two sequences $X = \{x_1, x_2, \dots, x_n\}$ and $Y = \{y_1, y_2, \dots, y_m\}$ with the same or different lengths, a warping path W is an alignment between X and Y, involving one-to-many mappings for each pair of elements. The cost of a warping path is calculated by the sum of the cost of each mapping pair. Furthermore, a warping path contains three constraints: (1) endpoint constraint: The alignment starts at pair $(1,1)$ and ends at pair (N, M); (2) monotonicity constraint: The order of elements in the path for both X and Y should be preserved in the same, original order of X and Y, respectively; (3) Step-size constraint: The difference of index for both X and Y between two adjacent pairs in the path needs to be no more than one step. In other words, pair (x_i, y_j) can be followed by three possible pairs including (x_{i+1}, y_j), (x_i, y_{j+1}) and (x_{i+1}, y_{j+1}).

DTW is a distance measure that searches for the optimal warping path between two series. We first construct a cost matrix C, where each element $C(i,j)$ is a cost of the pair (x_i, y_j), specified by using a distance function. DTW is calculated based on dynamic programming. The initial step of DTW algorithm is defined as:

$$DTW(i,j) = \begin{cases} \infty \ if \ (i = 0 \ or \ j = 0) \ and \ i \neq j \\ 0 \qquad\qquad\qquad if \ i = j = 0 \end{cases} \quad (1)$$

The recursive function of DTW is defined as

$$DTW(i,j) = min \begin{cases} DTW(i-1,j) + w_h C(i,j) \\ DTW(i,j-1) + w_v C(i,j) \\ DTW(i-1,j-1) + w_d C(i,j) \end{cases} \quad (2)$$

where (w_h, w_v, w_d) are weights for the horizontal, vertical and diagonal directions, respectively. $DTW(i,j)$ denotes the distance or cost between two sub-sequences $\{x_1, x_2, \dots, x_i\}$ and $\{y_1, y_2, \dots, y_j\}$, and $DTW(N,M)$ indicates the total cost of the optimal warping path.

For example, as shown in Appendix A, in the two sequences $A = \{1,2,3,4,5\}$, $B = \{1,2,3,4,3,2,5\}$, we first constructed a distance matrix. To calculate the value of each cell, we followed a combination of formula (1) and (2) as below:

$$dtw(i,\ j) = \left|A_i - B_j\right| + \min\left(D[i-1, j-1], D[i-1, j], D[i, j-1]\right) \quad (3)$$

For example, to get the value in cell on Column 2 Row 5 (in reverse order), that is highlighted by a red box in Appendix A, we calculated $dtw(5,\ 2) = |5 - 2| + \min(6,3,10) = 3 + 3 = 6$. After the whole matrix is developed, the shortest path (highlighted in yellow) starting from the diagonal corner. We added up the shortest path to get the DTW distance similarity score between the two sequences as $3 + 3 + 1 + 0 + 0 + 0 + 0 = 7$.

It is noted that the DTW applies to numeric sequences, in which all the elements are numbers rather than categorical values in other sequence measures such as the longest common subsequence (He et al., 2021).

4.2 Sequence clustering

Similar to the employment of DTW method in He et al. (2023), we used the maximum value of the Silhouette index to determine the optimal number of clusters in the missing response pattern sequence clustering. The results showed the optimal number of six clusters with the highest Silhouette index; therefore, we decided to use six clusters in this study and report our findings thereafter.

Figure 10 exhibits the centroids of six sequence clusters representing the typical missing response patterns across the nine tasks in the Farm Drone module. It is noted that the combination score of efficiency and correctness is within a range of 0 to 4, with missing response labeled as 9. Cluster 1 took the biggest proportion (30%) in the sample, followed by Cluster 3 (22.9%), Cluster 5 (16.5%), Cluster 4 (16.4%), Cluster 2 (11.4%), and Cluster 6 (2.9%), which took the smallest proportion in the sample.

The missing response pattern in the first cluster can be interpreted in a straightforward way: typically, students tried to solve the first two items and were more likely to succeed in the first item but fail in the second item. Students in Cluster 1 skipped the following questions until item 7. The students also tried item 8 but usually failed, and then skipped the last item. Students in Cluster 2 showed a typical missing response pattern on item 7 and item 9, while they seldom skipped any item until the sixth task. Students also tried item 8 but failed in finding a good solution. Cluster 3 missed responses at the beginning and the end of the CT module, though the reasons for missing responses might be different. Missing a response at the beginning was more likely to be caused by misunderstanding of the item (e.g., taking the first two items regarded as trial items without scores, while skipping the last item might be caused by shortage of time towards the end of testlet). Students in Cluster 3 made good efforts in solving items located in the middle (i.e., item 3 to item 8). Students in Cluster 4 showed skipping behaviors after making great efforts in the first half of the module. The skipping might be caused by insufficient time to continue solving problems in the second half of the module or students just simply gave up solving the items after they made a self-evaluation that deemed their ability could not surpass the more complex computation requirements in item 6 and beyond. Cluster 5 and 6 presented similar patterns in missing responses. One of the important differences is the effort exerted in item 7. Students in Cluster 5 kept a consistent CT score from item 1 to item 7. Most of the time, students got the correctness score, but might not have found an efficient solution across the items. Comparatively, students in Cluster 6, got high efficiency scores in the middle of the module, and

even got the highest efficiency and correctness score (4 score point) in item 7. However, the great efforts in item 1 to item 7 might have taken too much time, hence the students might have rushed to the end.

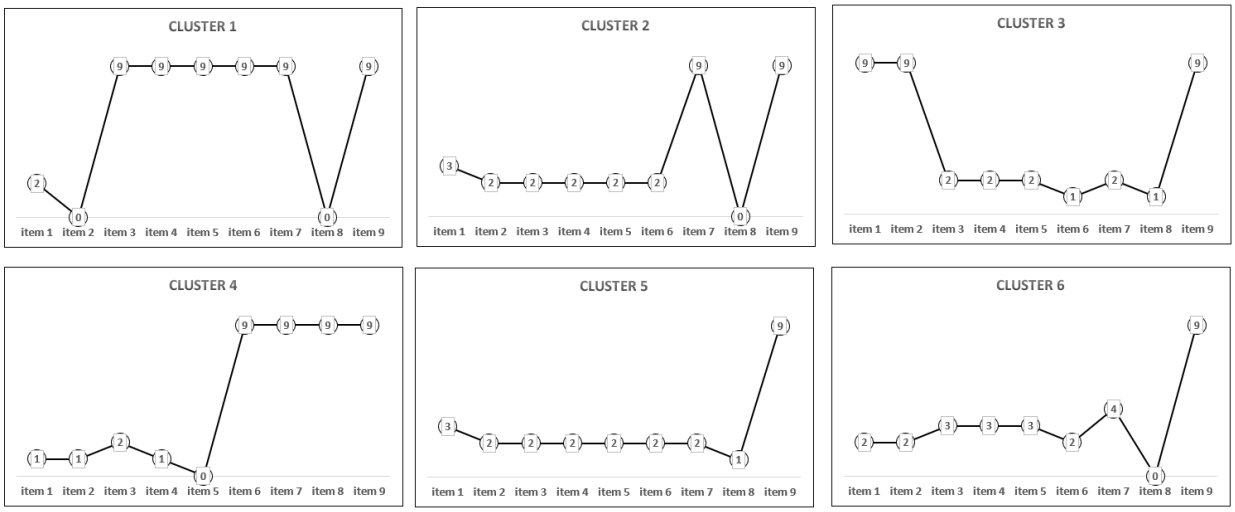

Figure 10. *Centroids of sequence similarity clustering for missing response patterns*

Figure 11 shows the sequence of time spent on each task by the six clusters, which provides further information to help pinpoint the potential reasons for the missing responses. In general, Cluster 1 solved all items much quicker than their peers. The fastest record (lower than 200 seconds) was found in item 4. Though students in Cluster 1 had the most missing responses compared with other patterns, the time they spent on the missing response items was not necessarily shorter. For instance, students in Cluster 1 even spent slightly more time than the other clusters on the last two items. This group of students spent a relatively equal amount of time on each item (except item 5), no matter if the items were difficult or easy. This suggested an aimless approach when solving the CT task. Cluster 2 and Cluster 6 showed similar time allocation patterns, though most of the missing responses occurred toward the end of the module. In these two clusters, students allocated too much time to complex items. They spent a significant amount of time on item 5 and item 7, especially item 7, but might not have had enough time to solve the last two items. These kinds of missing responses might have occurred because of insufficient time. In other words, if the students were given enough time to finish the tasks, they should have achieved high scores in efficiency and correctness given their previous performance in item 1 to item 7. Students in Cluster 3 and Cluster 5 followed a similar time allocation pattern. Students spent sufficient time on item 5, but made less effort on item 7, probably because little time remained, or because they were tired. Students in Cluster 4 typically spent a long time on item 5 and then decreased the time spent on all follow-up questions. This group of students might have regarded item 5 as very challenging to code. The occurrence of missing responses after item 5 might have been caused by a possible loss of interest or confidence in continuing the assessment.

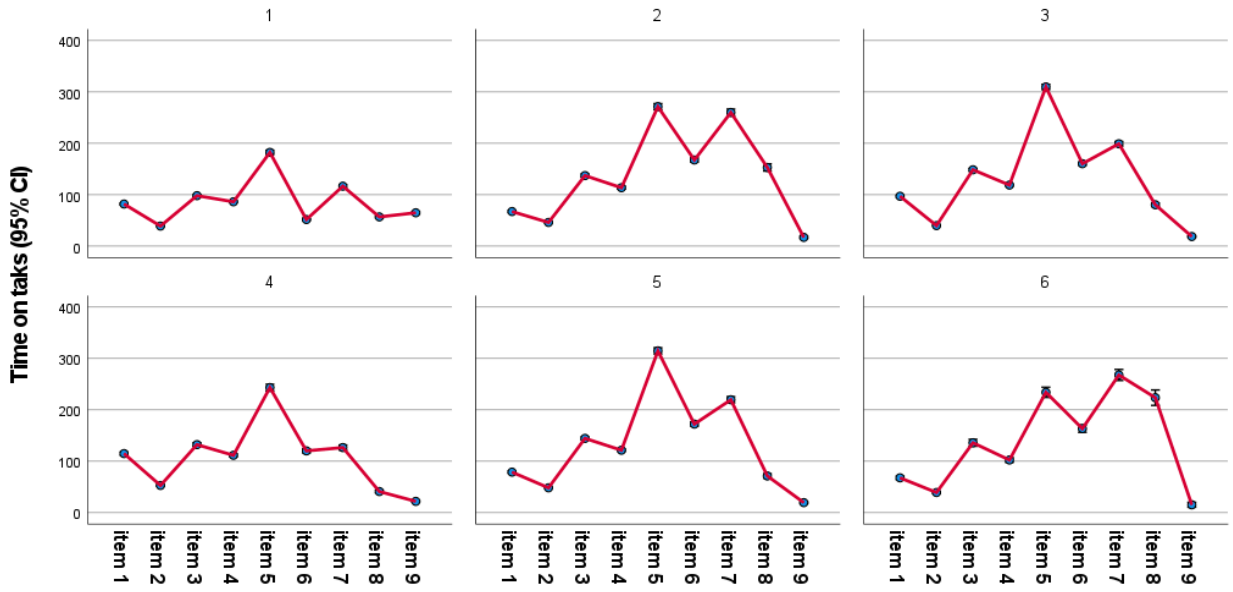

Figure 11. *Time on task by missing response patterns*

We further mapped the six missing response clusters onto the CT and CIL proficiency scale. The proficiency scores in both CT and CIL were found to be significantly different among the six clusters. On average, Cluster 6 had the highest scores for CT and CIL, followed by Cluster 2, Cluster 5, Cluster 3, Cluster 4, and Cluster 1. The results suggested that even though students in Cluster 6 and Cluster 2 did not give a full response to all the items, they still showed a very high level of ability according to their existent responses. As mentioned above, this group of students exerted great efforts and allocated more time in complex items (item 5 and item 7) and achieved the highest efficiency and correctness score in item 7 but might not have had enough time to finish the last one or two items. Students who showed missing response patterns at the beginning and the end of module (i.e., Cluster 3) showed a mid-level scores for CT and CIL. This group of students might not have got used to the environment at the beginning of the test and allocated too much time to the complex items. On average, students from Cluster 1 and Cluster 4 had low CT and CIL proficiency scores. This might be a result of them having missed many responses in the module, which could be a sign of low engagement, lack of targeted effort, fatigue, or vague understanding of programming code.

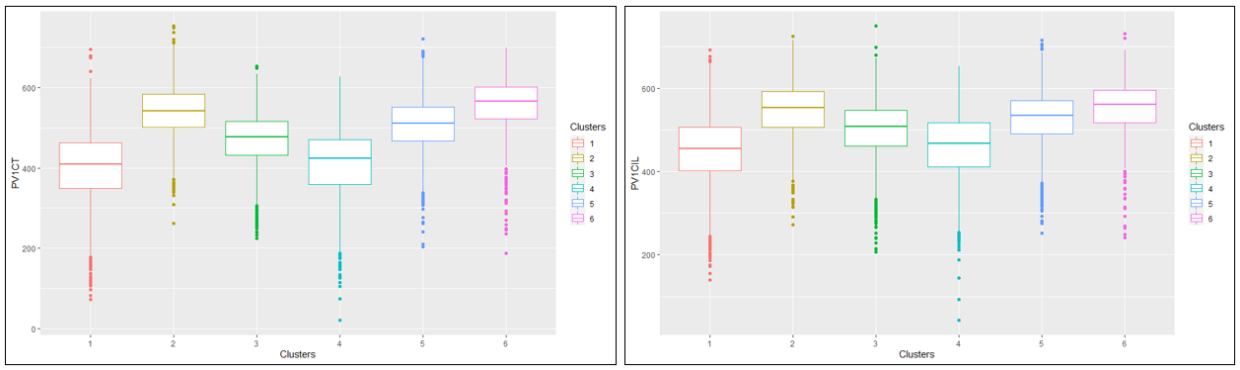

Figure 12. *CIL and CT proficiency by missing response patterns*

5. Discussion and Conclusion

This study is one of the pilot studies that focus on exploring computational thinking measurement such as ICILS with the use of process data. We employed K-Prototype clustering and DTW sequence clustering methods to identify test-takers' homogenous behavioral patterns and problem-solving strategies by separating the full response sample and samples with at least one missing response on one CT module, Farm Drone, with 9 programming tasks in total.

It was found that the use of the reset function, time allocation, length of commands, and algorithms (programming strategy) were all important factors to extract representative behavioral patterns. The use of the reset function could help students quickly go back to the default status and increase the chance of success in problem-solving. However, frequent use of the reset function did not necessarily correspond to a higher success rate or higher CT or CIL proficiency scores. Actually, too many reset clicks may indicate the opposite, i.e., students are more likely to be off track. Commands that are too short or too long may not be helpful in enhancing CT score. Efficiency in programming code is also an important factor that needs to be highlighted. Students who managed their time well enough to be able to solve all nine tasks in the module were more likely to get higher CT proficiency scores. Those who spent too much time on complex items (such as item 5 and item 7) might not have had enough time to complete the last several items. It was noticed that students in the Republic of Korea and France used the reset button more in Task 5 to Task 9. This finding might suggest possible differences in school instruction in these two countries, for example, teachers might share tips in their computational thinking or computer science class to instruct students to restart the coding process by using the reset function[5] to save programming and debugging time.

Given the high missing response rate in the ICILS sample in CT Farm Drone module, about two-thirds of students had at least one missing response, it would be wise to make a separate study to extract the missing response patterns and understand when, where, and why the

[5] There are two reset icons on the task interface. One reset icon controls the restarting farm drone simulation while the other reset icon controls the resetting of workspace (i.e., cleaning up all program codes in the workspace or return to pre-populated default codes in debugging and configuration tasks. The process data only captured the clicks on the reset icon in the simulation environment, therefore the frequency of reset, in this study, indicates the clicks on reset icon in simulation only.

missing responses might occur. The time allocation on the complex items needs to be wisely arranged. Though as our expectation, most missing responses were found in the last two items within the module. Item 5 would be a key item to double check. This middle-located item for the first time introduced more complex requirements for the farm drone from one single row to multiple rows. This confused many students, which was shown in the response time distribution and high missing response rate just after item 5. Students who exerted great efforts in solving programming tasks, but did not manage their time well through to the end, were still found to be highly proficient. For students who skipped items after the middle key item (item 5), their proficiency scores were lower than their peers.

Some limitations also merit discussion. First, in ICILS, it was challenging to use process data on an individual level, but only aggregate level. The lack of granular process data may not be rich enough to do deeper finer-grained level research. We would highly recommend IEA to consider capturing more process data (at least action sequences and timestamp) to support related research in the future.

Second, in the current research, we focused more on CT and CIL proficiency scores, rather than background variables. It would make better sense to compare the clustering result by gender, socioeconomic status, immigrant status, and survey questions related to programming coding.

Third, the loose information extracted from process data calls for better documentation and storage. Some suggestions for the ICILS process data framework: (1) capture action sequence drag and drop, change, and time stamp, this is very important for tracking the students' coding strategy; (2) capture the jump back and forth, and transition between items; (3) the reset tracking function needs to separate the reset on the coding page from the reset on the animation page; and (4) remaining time needs to distinguish between students who complete all items but still remain in time and students who skip items but still have time remaining.

Finally, the clustering results have not yet been connected with theory and are still in the descriptive stage. Especially in study 1, the interpretation of the four clusters of behavioral patterns is based on the qualities/features of each attribute, but there is not yet a comprehensive explanation of why these patterns occur in four clusters, or of the link between CT skills and these homogenous patterns.

The focus of this study could be expanded in the future from descriptive analysis to further measurement modeling. For example, study 2 captured the pattern of missing responses across the unit, which helped us to better understand the various reasons for missed responses and helped us to group these into different categories. This approach could help us to categorize the missing responses more efficiently and measure the latent skills more accurately. There is an association between the missing response patterns and CT skills in study 2, which means we could calibrate the difficulty of each item separately for each group and estimate the most appropriate parameters for each interactive task. A future study could use behavioral variables to better predict the missing responses by incorporating both peers' pattern and individual's special sequence pattern.

In summary, this study provides a new angle on the measurement of students' CT and CIL skills with the utilization of process data. Aside from the limited information captured in ICILS 2018, other researchers could explore new methods for handling process data.

Advanced process data analysis either by variable-based approach, machine learning techniques, or a combination of the two, would be worth extensive further research in the near future.

Acknowledgement

This project is financially supported by IEA Research and Development Award Funding. The authors thank Julian Fraillon of IEA and Tim Friedman of ACER for their help in a variety of technical support. The authors also thank Ralph Carstens, Lauren Musu, Raegen Jackson of IEA for their efficient project management and guidance, and thank Paula Korsnakova and Dr. Dirk Hastedt for their helpful suggestions in IEA research projects. The authors also thank Dr. Jeppe Bundsgaard for constructive discussion in the project preparation phrase.

References

Amin, T. B., & Mahmood, I. (2008). Speech recognition using dynamic time warping. *2008 2nd International Conference on Advances in Space Technologies,* 74–79. IEEE. https://doi.org/10.1109/icast.2008.4747690

Anderberg, M. R. (1973). The broad view of cluster analysis. *Cluster analysis for applications, 1*(1), 1–9. https://doi.org/10.1016/C2013-0-06161-0

Chen, Y., Li, X., Liu, J., & Ying, Z. (2019). Statistical analysis of complex problem-solving process data: An event history analysis approach. *Frontiers in Psychology*, *10*, 486. https://doi.org/10.3389/fpsyg.2019.00486

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM, 60*(6), 33–39. https://dl.acm.org/doi/10.1145/2998438

Dong, G., & Pei, J. (2007). *Sequence data mining*. Springer Science & Business Media. https://doi.org/10.1007/978-0-387-69937-0

Fraillon, J., Ainley, J., Schulz, W., Duckworth, D., & Friedman, T. (2019). *IEA International Computer and Information Literacy Study 2018 assessment framework.* Springer, Cham . https://doi.org/10.1007/978-3-030-19389-8

Fraillon J., Ainley, J., Schulz, W., Friedman, T., & Duckworth, D. (2020). *Preparing for life in a digital world—IEA International Computer and Information Literacy Study 2018 international report*. Springer, Cham. https://doi.org/10.1007/978-3-030-38781-5

Gabadinho, A., Ritschard, G., Müller, N. S., & Studer, M. (2011). Analyzing and visualizing state sequences in R with TraMineR. *Journal of Statistical Software, 40*(4), 1–37. https://doi.org/10.18637/jss.v040.i04

Gao, Y., Cui, Y., Bulut, O., Zhai, X., & Chen, F. (2022). Examining adults' web navigation patterns in multi-layered hypertext environments. *Computers in Human Behavior, 129*, 107142. https://doi.org/10.1016/j.chb.2021.107142

Goldhammer, F., Naumann, J., Rölke, H., Stelter, A., & Tóth, K. (2017). Relating product data to process data from computer-based competency assessment. In D. Leutner, J. Fleischer, J. Grünkorn, E.Klieme (Eds.), *Competence assessment in education* (pp. 407–425). Springer, Cham. https://doi.org/10.1007/978-3-319-50030-0_24

Guo, A., & Siegelmann, H. T. (2004). Time-warped longest common subsequence algorithm for music retrieval. *ISMIR*. https://groups.cs.umass.edu/binds/wp-content/uploads/sites/21/2019/05/2004_AnYuan_ISMIR.pdf

Hamming, R. W. (1950). Error detecting and error correcting codes. *The Bell System Technical Journal, 29*(2), 147–160. https://doi.org/10.1002/j.1538-7305.1950.tb00463.x

Han, Z., He, Q., & von Davier, M. (2019). Predictive feature generation and selection using process data from PISA interactive problem-solving items: An application of random forests. *Frontiers in Psychology, 10*. https://doi.org/10.3389/fpsyg.2019.02461

Hao, J., Shu, Z., & von Davier, A. (2015). Analyzing process data from game/scenario-based tasks: An edit distance approach. *Journal of Educational Data Mining, 7*(1), 33–50. https://doi.org/10.5281/zenodo.3554706

He, Q., Borgonovi, F., Suárez-Álvarez, J. (2023). Clustering sequential navigation patterns in multiple-source reading tasks with dynamic time warping method. *Journal of Computer-Assisted Learning, 39*(3), 719–736. https://doi.org/10.1111/jcal.12748

He, Q., Borgonovi, F., & Paccagnella, M. (2019). Using process data to understand adults' problem-solving behaviour in the Programme for the International Assessment of Adult Competencies (PIAAC): Identifying generalised patterns across multiple tasks with

sequence mining. *OECD Education Working Papers, No. 205*, OECD Publishing. https://doi.org/10.1787/650918f2-en

He, Q., Borgonovi, F., & Paccagnella, M. (2021). Leveraging process data to assess adults' problem-solving skills: Identifying generalized behavioral patterns with sequence mining. *Computers and Education, 166,* 104170. https://doi.org/10.1016/j.compedu.2021.104170

He, Q., & von Davier, M. (2015). Identifying feature sequences from process data in problem-solving items with n-grams. In A. van der Ark, D. Bolt, S. Chow, J. Douglas & W. Wang (Eds.), *Quantitative psychology research: Proceedings of the 79th annual meeting of the Psychometric Society* (pp.173–190). Springer. https://doi.org/10.1007/978-3-319-19977-1_13

He, Q., & von Davier, M. (2016). Analyzing process data from problem solving items with n-grams: Insights from a computer-based large-scale assessment. In Y. Rosen, S. Ferrara, & M. Mosharraf (Eds.), *Handbook of research on technology tools for real-world skill development* (pp. 750–777). IGI Global. http://dx.doi.org/10.4018/978-1-4666-9441-5.ch029

Heldt, M., Massek, C., Drossel, K., & Eickelmann, B. (2020). The relationship between differences in students' computer and information literacy and response times: An analysis of IEA-ICILS data. *Large-scale Assessments in Education, 8*(1), 1–20. https://doi.org/10.1186/s40536-020-00090-1

Hirschberg, D. S. (1975). *The longest common subsequence problem.* Princeton University.

Hirschberg, D. S. (1977). Algorithms for the longest common subsequence problem. *Journal of the ACM (JACM), 24*(4), 664–675. https://doi.org/10.1145/322033.322044

Huang, Z. (1997). A fast clustering algorithm to cluster very large categorical data sets in data mining. Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, Dept. of Computer Science, The University of British Columbia, Canada (pp. 1-8). https://www.diag.uniroma1.it/~sassano/STAGE/Fast_Clustering.pdf

Huang, Z. (1998). Extensions to the K-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery, 2*(3), 283–304. https://doi.org/10.1023/A:1009769707641

Jurafsky, D. & Martin, J.H. (2008). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.* Prentice Hall.

Kurbalija, V., Radovanović, M., Geler, Z., & Ivanović, M. (2011). The influence of global constraints on DTW and LCS similarity measures for time-series databases. In D. Dicheva, Z. Markov, E. Stefanova (Eds.), *Third International Conference on Software, Services and Semantic Technologies S3T 2011* (pp. 67–74). Springer. https://doi.org/10.48550/arXiv.1107.0134

Launeanu, M., & Hubley, A. M. (2017). Some observations on response processes research and its future theoretical and methodological directions. In B. Zumbo & A. Hubley (Eds.), *Understanding and investigating response processes in validation research* (pp. 93–113). Springer, Cham.

Levenshtein, V. (1965). Binary codes capable of correcting spurious insertions and deletions of ones. *Russian Problemy Peredachi Informatsii, 1*, 12–25.

Levenshtein, V. I. (1966, February). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady, 10*(8), 707–710.

Liao, D., He, Q., & Jiao, H. (2019). Mapping background variables with sequential patterns in problem-solving environments: An investigation of U.S. adults' employment status in PIAAC. *Frontiers in Psychology, 10*, 646. https://doi.org/10.3389/fpsyg.2019.00646

MacQueen, J. (1967, June). Some methods for classification and analysis of multivariate observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 281–297. http://www.cs.cmu.edu/~bhiksha/courses/mlsp.fall2010/class14/macqueen.pdf

Permanasari, Y., Harahap, E. H., & Ali, E. P. (2019). Speech recognition using dynamic time warping (DTW). *Journal of Physics: Conference Series, 1366*(1), 012091. https://doi.org/10.1088/1742-6596/1366/1/012091

Qiao, X., & Jiao, H. (2018). Data mining techniques in analyzing process data: A didactic. *Frontiers in Psychology, 9*, 2231. https://doi.org/10.3389/fpsyg.2018.02231

Ren, Z., Fan, C., & Ming, Y. (2016, November). Music retrieval based on rhythm content and dynamic time warping method. *2016 IEEE 13th International Conference on Signal Processing (ICSP),* 989–992. IEEE. https://doi.org/10.1109/ICSP.2016.7877977

Sahin, F., & Colvin, K. F. (2020). Enhancing response time thresholds with response behaviors for detecting disengaged examinees. *Large-scale Assessments in Education, 8*(1). https://doi.org/10.1186/s40536-020-00082-1

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 26*(1), 43–49. https://doi.org/10.1109/TASSP.1978.1163055

Salles, F., Dos Santos, R., & Keskpaik, S. (2020). When didactics meet data science: Process data analysis in large-scale mathematics assessment in France. *Large-scale Assessments in Education, 8*(1). https://doi.org/10.1186/s40536-020-00085-y

Studer, M., & Ritschard, G. (2014). A comparative review of sequence dissimilarity measures. *LIVES Working Papers, 33*, 1–47. https://doi.org/10.12682/lives.2296-1658.2014.33

Sukkarieh, J. Z., von Davier, M., & Yamamoto, K. (2012). From biology to education: Scoring and clustering multilingual text sequences and other sequential tasks. *ETS Research Report No. RR-12-25*. . https://doi.org/10.1002/j.2333-8504.2012.tb02307.x

Tang, X., Wang, Z., He, Q., Liu, J. & Ying, Z. (2020). Latent feature extraction for process data via multidimensional scaling. *Psychometrika, 85*(2), 378–397. https://doi.org/10.1007/s11336-020-09708-3

Ulitzsch, E., He, Q., & Pohl, S. (2022). Using sequence mining techniques for understanding incorrect behavioral patterns on interactive tasks. *Journal of Educational and Behavioral Statistics, 47*(1), 3–35. https://doi.org/10.3102/10769986211010467

Ulitzsch, E., He, Q., Ulitzsch, V., Nichterlein, A., Molter, H., Niedermeier, R., & Pohl, S. (2021). Combining clickstream analyses and graph-modeled data clustering for identifying common response processes. *Psychometrika*, *86*(1), 190–214. https://doi.org/10.1007/s11336-020-09743-0

Appendix A

An example of distance matrix computed by dynamic timing warping method

A	**5**	10	6	3	1	2	4	3
	4	6	3	1	0	1	3	3
	3	3	1	0	1	1	2	4
	2	1	0	1	3	4	4	7
	1	0	1	3	6	8	9	13
		1	**2**	**3**	**4**	**3**	**2**	**5**
		B						

Note. This example is to compute the distance similarity measure between two sequences: $A = \{1,2,3,4,5\}$ and $B = \{1,2,3,4,3,2,5\}$. The highlighted yellow path is the shortest path starting from the upper right diagonal corner to the lower left diagonal corner. The sum of value along the highlighted cells is the similarity score between sequences A and B.