Hindawi
Scientific Programming
Volume 2022, Article ID 3575457, 15 pages
https://doi.org/10.1155/2022/3575457

Research Article

A Novel Three-Way Decision Model for Improving Computational Thinking Based on Grey Correlation Analysis

Ruiyang Xu, Chunmao Jiang, and Lijuan Sun

School of Computer Science and Information Engineering, Harbin Normal University, Harbin, China

Correspondence should be addressed to Chunmao Jiang; chunmaojiang@hrbnu.edu.cn

Received 18 November 2021; Accepted 4 January 2022; Published 25 January 2022

Academic Editor: Rahman Ali

Computational thinking (CT) is an approach that applies the fundamental concepts of computer science to solve problems, design systems, and understand human behavior, which can help students develop lifetime learning and generate new topics. It has been the elements of competency expected of the next generation of talents. However, the current research on computational thinking evaluation is still at a relatively weak stage. The existing related evaluation research is still limited to traditional curriculum evaluation methods. Therefore, the training effect of computational thinking cannot be well quantified, and the characteristics of students cannot be further explored. In this work, we propose a three-way decision model for improving computation thinking. We first developed a system of evaluation metrics, including five specific primary indicators and several secondary indicators. Next, the weight of each indicator was determined by applying an expert similarity measure, consequently getting the best metric sequence. We employ a grey correlation analysis to calculate the distance of each test result from this optimal sequence. Then, we trisect the set of testers based on the distance to build three regions of high score sequences, medium score sequences, and low score sequences inspired by the three-way decision. We can then exploit these rules on target students in the relatively low regions to improve their computational thinking. An example analysis illustrates the effectiveness and applicability of the method. This article provides a solid theoretical basis for improving students' computational thinking ability. Teaching administrators can conveniently formulate computational thinking teaching strategies, and timely warning and intervention for students with poor computational thinking ability can effectively improve students' computational thinking ability. The corresponding training measures are given to students of different ability levels to achieve differentiated and personalized training.

1. Introduction

With the rapid development of artificial intelligence and information technology, human thinking is experiencing change, and computational thinking has become essential in the information age. As a new method of the intelligent information age, computational thinking is a kind of thinking activity that can flexibly use computational tools and strategies to solve problems. The cultivation of computational thinking can promote the comprehensive development of people and benefit a lifetime.

Computational thinking has attracted widespread attention in international primary education since it was proposed in 2006 [1], and curriculum standards related to computational thinking have been developed. The U.S. Computer Science Standards (CSTA) for grade K12, published in 2011, has included computational thinking as a critical element of the computer science curriculum. The British Ministry of Education issued the Computational Learning Plans I–IV in 2013 to guide the development of computational thinking skills for students in primary education in the UK. In 2015, the Australian Ministry of Education released the Digital Technology Curriculum Standards, emphasizing that people need computational thinking literacy in a digital information society. China also gradually pays attention to the development of computational thinking education. In 2010, the C9 University Consortium emphasized that developing computational thinking skills would be a significant, long-term, and complex core task of primary computer teaching. In 2012,

the Ministry of Education made the cultivation of computational thinking a priority. It pushed the reform of the computer curriculum to improve the practical application of computers and realize computer empowerment education. In 2017, it included computational thinking in the General High School Information Technology Curriculum Standards as one of the four core elements of the subject.

Thus, computational thinking education has gradually become younger, and more teachers and parents are paying more and more attention to cultivating computational thinking skills from a young age. It has steadily practiced the cultivation model of computational thinking and teaching strategies, but how effective is the cultivation? Are students' computational thinking skills improved, and how are they being evaluated? Without reliable assessment tools or methods, it is not easy to make the best use of computational thinking when it is integrated into educational curricula. Evaluation is crucial for developing computational thinking and is a prerequisite for developing student's computational thinking skills. Pedagogical evaluation is a guide for developing computational thinking and a guarantee of its sustainability.

Only by fully understanding the shortcomings in the development of computational thinking, we can design a scientific, reasonable, and perfect assessment system in a targeted manner, thus well-developing student's computational thinking skills. Computational thinking evaluation research is still in its infancy. There is still a lack of professional evaluation systems and evaluation methods that can quantify the effect of developing computational thinking. Combining teaching practice with quantitative evaluation of student's computational thinking ability is the next question researchers must consider. We can only facilitate the research of the following cultivation strategies by fully grasping student's computational thinking ability. Therefore, a scientific and reasonable teaching evaluation will have a decisive influence on the cultivation of computational thinking. By evaluating student's computational thinking skills, it is possible to grasp student's abilities and thus give different training strategies to students with unique characteristics, thus meeting society's demand for individualized talents. The evaluation results can explore the features of students with varying levels of ability and then give corresponding training to students with different levels of ability, thus achieving differentiated and individualized training. Therefore, a reasonable evaluation model and ability feature mining research are significant for student's personalized computational thinking.

We organized the rest of this study as follows. In Section 2, we review the strategies for developing computational thinking and measuring computational thinking. Section 3 proposes a computational thinking evaluation metric framework. We employ the grey correlation between the comparative sequence of the test taker and the optimal reference sequence to construct three regions of high level, medium level, and low level, according to two thresholds by sorting them according to the correlation value. In Section 4, we perform a three-way classification and determine the final category. Then, the hidden association rule properties behind the student evaluation results are mined based on the Apriori algorithm. Section 5 gives a summary and planning for future work.

2. Related Work

This section will review computational thinking and its related evaluation methods and then review methods such as grey correlation analysis and three-way decision.

2.1. Computational Thinking Development Strategies. Robotics and programming are crucial vehicles and avenues for the development of computational thinking. Angeli and Valanides [2] studied the effect of educational robots on student's computational thinking of different genders. The results showed that boys benefited more from spatial orientation and manipulative activities, while girls benefited more from collaborative writing activities. This research contributes to the body of knowledge about teaching computational thinking. The results can design lessons and classroom activities that focus on a broader range of computational thinking skills. Chalmers [3] studied how Australian elementary school teachers integrated robotics and coding in their classrooms and its impact on student's computational thinking skills. The results showed that using robotic tools and activities for exploration can help teachers build confidence and a body of knowledge. Relkin [4] et al. studied changes in computational thinking skills in the first- and second-grade students. The results provide that teaching young children to code can accelerate their computational thinking skills. Özmutlu [5] et al. studied the impact of short-term, intensive coding and robotic training on the self-efficacy of middle school students' computational thinking skills.

There would be many possibilities to explore the impact of these experiences on elementary and students in the areas of coding, robotics, mobile devices, Arduino-based applications, and game-based learning. Gadzikowski [6] designed coding, robotics, and engineering course for young students to learn knowledge, such as coding, robotics and engineering concepts, and practice skills, such as creative problem-solving, computational thinking, and critical thinking. Qu and Fok [7] focused on student-robot interactions in robotic education and attempted to cultivate student's computational thinking skills. Chevalier et al. [8] discussed how educational robotics fostered computational thinking skill development and confirmed that robotic education is necessary for specific teaching interventions.

Xiao and Yu [9] explored teaching computational thinking in four stages one by one, from problem identification and decomposition, system abstraction and solution design optimization, solution implementation, and problem migration, with an engineering design perspective of problem-solving. Vesikivi et al. [10] focused on teaching computational thinking and the teaching methods and research design under different types on the impact of the development of computational thinking. Cui and Ng [11] studied evidence-based directions towards enriching mathematics education with computational thinking. Grover et al. [12] tapped into the existing relationship

between cognitive level and computational thinking through student's programming behaviors, thus showing the superiority of programming instruction as a means of computational thinking development. Based on computational review and app inventor characteristics, Ku [13] proposed developing student's computational thinking skills with the teacher as the designer, organizer, guide, and app inventor learning tool. The method motivates students to actively use computational thinking to analyze and solve problems through teacher-student cooperation and student-student cooperation as learning forms.

2.2. Computational Thinking Evaluation Methods. Existing computational thinking evaluation methods include programming task-based assessments [14, 15] and scale assessments [16–19]. Automatic scoring systems based on programming tasks automatically score the test taker's computational thinking skills by the learner's programming code situation. For example, an automatic scoring system based on programming tasks automatically scores the test taker's computational thinking ability based on the learner's programming code. Another approach to programming-based present assessment is the design of a computational thinking assessment framework, which evaluates programming items based on the computational thinking concepts, practices, and perspectives involved in the programming project.

The scale assessment methods include the test-based evaluation scale CT, which assesses computational thinking through actual student project answers. There are also evaluation scales based on the five factors of computational thinking designed to evaluate student's computational thinking based on their behavioral data and an evaluation scale based on self-efficacy, which evaluates the learners' level of computational skills. Román-González et al. developed a multi-competency test-based evaluation scale, CTt, to assess the computational thinking ability of the subjects. Korkmaz et al. used the theoretical framework of computational thinking proposed by ISTE as a basis to design the computational thinking scale (CTS). In 2015, Korkmaz et al. similarly designed and oriented the scale to measure college student's level of computational thinking skills, which comprised 5 factors and 29 measures, and validated the reliability and validity of the computational thinking scale. It was later revised by Korkmaz et al., and the scale was oriented to students at the K12 level. The revised CTS still contains the original five factors and 22 measures with the same validity and reliability and focuses on measuring different age groups. Kukul et al. developed the computational thinking self-efficacy scale (TSES), through which learners self-assess their level of computational competence. Brennan et al. proposed a three-dimensional evaluation framework and argued that assessment can be carried out in terms of the concepts (e.g., sequence, loop, and parallelism), practices (e.g., incremental and iterative, testing and debugging, and reuse and recreation), and perspectives (e.g., expression, communication, and questioning) of computational thinking.

The above assessment methods collect student data and scores based on items or scales. The subjective scoring of learners for each task based on teachers' experience is, first, more subjective and, second, does not consider the evaluation index levels and index weights. The simple statistical method of measuring the effect of computational thinking training cannot tap into the deep relationships among students, which is not conducive to proposing targeted training strategies. Analyzing student's data and exploring the hidden relationships between student's computational thinking levels are an urgent problem to be solved. The three-way classification has been extensively investigated and applied in various situations.

2.3. Three-Way Decision and Three-Way Classification. With the rapid development of massive data and artificial intelligence, decision-making has become increasingly prominent [20, 21]. Instead of the traditional binary classification problem, Yao first outlined a three-way decision theory [22], applied to the classification problem by "thinking in three." The third alternative of the boundary region is introduced, which is associated with deferred or indeterminacy decisions of the classification. The three-way classification has been extensively investigated and applied in various situations [23].

The trisecting-acting-outcome (TAO) model of a three-way decision encompasses three components: trisecting divides a whole into three pairwise disjoint or weakly joint regions P_1, P_2, and P_3. The acting is to devise action strategies for three regions. The outcome evaluation measures the effect of the trisection and action strategy [24–26]. The TAO model has merged as a new three-way decision model that promises to make the three-way decision smarter. The major concern regarding the TAO model is about the outcome that is the effectiveness of trisecting and acting.

Using three-way classification in developing teaching strategies is a significant experiment. The framework for measuring and improving the level of computational thinking using three-way decision, especially the TAO model in this study, is illustrated in Figure 1.

All students are assessed from a whole, as shown in "A whole," based on specific multilevel metrics, which may be two levels. Thus, we made three segments: high-level, medium-level, and low-level regions. Students in the high-level area have better marks on particular measures, while students in the low-level area have worse impacts on specific criteria. Analyzing these specific characteristics allows teachers to design customized instructional strategies to further develop student's specific competencies to improve their computational thinking. These instructional strategies form the "strategies" node. We can obtain the benefits of these two processes through the "outcome evaluation."

3. Evaluation of Computational Thinking Using Three-Way Decision

This section first proposes a computational thinking evaluation index system. It assigns weights to each index through expert clustering, which fully reflects the contribution of different experts to the index weight and avoids the disadvantages of

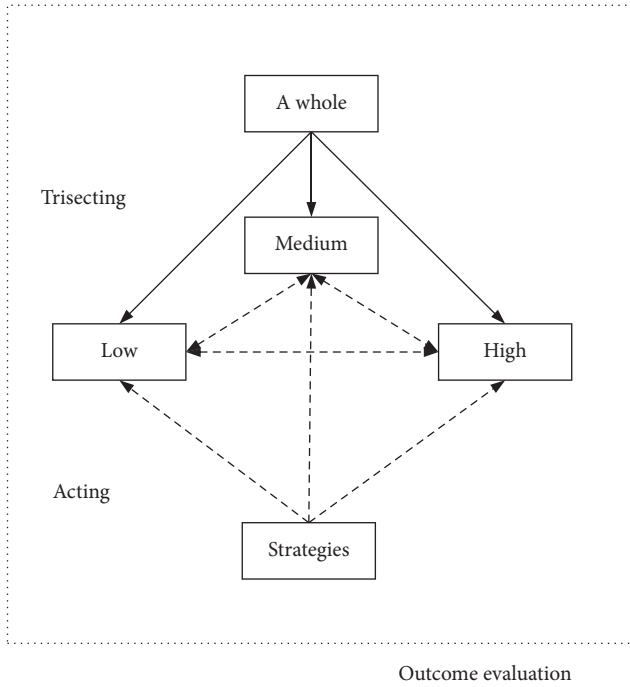

Figure 1: TAO model of trisecting-acting-outcome.

being too single subjective. Then, through the weighted grey correlation analysis method, the grey correlation degree between the comparison sequence of each testee and the optimal reference sequence is thoroughly studied and analyzed. The degree of correlation is sorted according to the value of the correlation degree. According to the set threshold, the students can be initial classification. The three regions of high level, medium level, and low level were constructed according to two thresholds by sorting them in order according to the correlation value.

3.1. Computational Thinking Evaluation Metric Framework. Computational thinking has different components, according to various scholars and research institutions. MIT's NEET program considers computational thinking to apply fundamental computational procedures, data structures, and algorithms to other social systems, such as production and life. Özgen considers computational thinking as a piece of knowledge, skills, and attitudes that enables computers to solve real-life problems. The British School Computing Curriculum Working Group [27] considers that the elements of computational thinking include logical, algorithmic, recursive, and abstraction skills. Brennan and Resnick [28] think that computational thinking comprises three major components: computational concepts, computational practices, and computational viewpoints, containing 16 areas of skills. Settle and Perkovic proposed a conceptual framework of computational thinking from the perspective of computer principles. ISTE believes that computational thinking comprises five components: creativity, algorithmic thinking, critical thinking, problem-solving, and collaboration. Selby and Woollard [29] argue that computational thinking comprises decomposition, abstraction, generalization, algorithm, and evaluation. Korkmaz et al. [17] argued that computational thinking includes cognitive and application-based knowledge structures related to computer science, e.g., problem representation and solving, and abstraction. Many researchers have continuously explored and refined computational thinking.

Since the concept was put forward from computational thinking, there has been a lot of research on the interpretation of the connotation of computational thinking and teaching. There has been a lot of research on the interpretation of the connotation of computational thinking, teaching methods, models, etc. However, there are relatively few studies on the evaluation of computational thinking. The existing evaluation methods of computational thinking include table evaluation method, work analysis evaluation method, interview evaluation method, question evaluation method, and evaluation of related computational thinking. However, most of these evaluation methods focus on simple score evaluation, and the evaluation indicators are not. Students' computational thinking characteristics behind these achievements are not deeply explored. Thus, the weighted establishment of evaluation indicators and the classification and mining of student characteristics have become the focus of this article. At the same time, the traditional student classification method classifies students as good or poor according to their rank or total proportion. It does not consider the relationship between multiple constituent indicators and the hierarchical relationship. In particular, for classifying middle school students, there is a problem of inaccurate classification, and the problem of inaccurate implementation of teaching strategies that follow brings additional teaching costs.

We integrate the five significant elements to design a computational thinking level evaluation metric based on the principles, including scientificity, feasibility, comprehensiveness, and independence. It incorporates the evaluation characteristics of programming education from the connotation and components of computational thinking and takes programming as a fundamental approach to cultivate computational thinking.

This study's evaluation metric framework of computational thinking contains five first-level indicators, including problem decomposition, abstraction, pattern generalization, algorithm, and evaluation. Moreover, on this basis, more fine-grained two-dimensional metrics are established to build a hierarchy of computational thinking evaluation metrics, as shown in Figure 2.

3.1.1. Second-Level Evaluation Metrics. The core of computational thinking is the logical decomposition of significant problems, thus breaking them down into smaller modules that are easier to solve. The indicator "decomposition" (denoted as U_1) includes two secondary measures: the ability to analyze the material studied (denoted as u_{11}) and the ability to decompose the problem (denoted as u_{12}). The former refers to understanding the material, organizing and analyzing it logically, and clarifying the problem's core. The latter refers to decomposing complex problems into more minor problems, clarifying the relationships between the smaller problems, and establishing a logical sequence of the different parts.

"Abstraction" (denoted as U_2) refers to extracting core things or critical data from many transactions and ignoring irrelevant details. The final representation in a formal way is the transformation of data or problems into a data structure or formal mathematical model suitable for computer processing. It comprises three secondary metrics: conceptual analysis (denoted as u_{21}), inductive extraction (denoted as u_{22}), and formal representation (denoted as u_{23}). Conceptual analysis refers to the ability to clarify the various concepts contained in a transaction and to clarify each concept and the relationship between concepts by means of comparison, judgment, and reasoning; inductive extraction refers to the ability to extract the common essential properties, methods, and rules of different things and then to exclude the nonessential parts or irrelevant details of the individuality of specific things. Formal representation refers to the representation of a problem so that a computer can solve it, thus forming an abstract representation and a visual representation of the object.

"Pattern" (also called a generalization, denoted as U_3) is a general pattern for solving a class of problems. It is used to summarize some specific problem-solving patterns by continuously comparing abstraction and generalization of problems and extending them to the solution of similar problems. It includes three secondary metrics: model construction (denoted as u_{31}), structural specification (denoted as u_{32}), and stable operability (denoted as u_{33}). The model construction shows the ability to summarize a pattern through the current problem, clarify the type of pattern, and be familiar with the things to be solved so that it can be applied to the same type of things. The structure specification means that the structure of the pattern is hierarchical and logical. The elements represented by the pattern are simple and can reflect the core and essence. Stable operability indicates that it can be appropriately applied to similar problems by simple modifications and has high applicability.

"Algorithm" (denoted as U_4) is a series of computer instructions for solving a problem, a collection of infinite rules. Algorithmic thinking and computer systems can form a series of automated solutions to problems. It consists of four secondary indicators: data representation (denoted as u_{41}), functional refinement (denoted as u_{42}), straightforward process (denoted as u_{43}), programming (denoted as u_{44}), and debugging, respectively. Data representation means that variables can be extracted, their type can be determined, and the relationships between the data can be analyzed. Finally, the appropriate data structure was chosen according to the needs of the problem. Function refinement means clarifying the program's specific functions, sorting out the logical relationships between functions, and defining different functions. Clarifying the flow means that a suitable structure can be built with flowcharts. Programming and debugging mean choosing the proper statements, translating the problem into a program, and debugging the errors to build a well-readable program.

"Evaluation" (denoted as U_5) is the process of using practical steps and resources to arrive at the most appropriate and suitable solution, procedure, or algorithm, by weighing the pros and cons and finding an ideal solution that is most applicable. It consists of three secondary metrics, namely completion (denoted as u_{51}), process optimization (denoted as u_{52}), and usability (denoted as u_{53}). Completion indicates whether or not the basic functionality can be accomplished as required and allows the correctness of the solution to be assessed. Program optimization refers to optimizing the program to make it functionally richer. Usability means that the program has a certain level of usability or better performance.

3.2. Establishing Three Partitions Based on Weighted Grey Correlation Analysis. Since the importance of metrics is different, it is necessary to distinguish the role of each metric in the overall evaluation, and determining the weight of metrics is one of the core issues of evaluation. In this section, the cosine similarity among experts completes the expert clustering. The intra-class weights and interclass weights of experts are based on the information entropy and the ratio of clustering numbers, respectively. Finally, the proportion of each expert is calculated comprehensively. Then, the final metric weights are obtained by combining the multiplicative sum of the initial value of each expert-rated metric and the proportion of each expert. This method can reflect the individual expert weights and comprehensively consider each expert's contribution to the index weights.

3.2.1. Weighting Analysis Based on Expert Clustering. Evaluation of computational thinking evaluates the combined effect of multiple factors rather than a single evaluation. On the basis of the evaluation metric framework for computational thinking in Figure 2, we derived a final

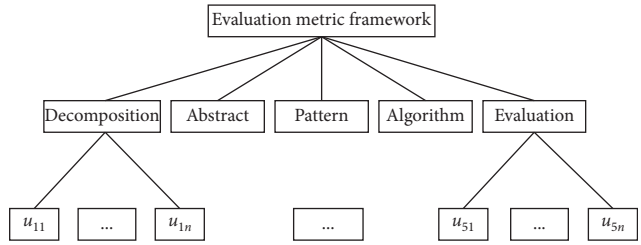

Figure 2: Evaluation metric framework.

evaluation for each student and divided this evaluation into three subdivisions, i.e., high-level area, medium-level area, and low-level area. The collection of the metrics includes the following:

$$\begin{aligned} U_1 &= \{u_1, u_2\}, \\ U_2 &= \{u_3, u_4, u_5\}, \\ U_3 &= \{u_6, u_7, u_8\}, \\ U_4 &= \{u_9, u_{10}, u_{11}, u_{12}\}, \\ U_5 &= \{u_{13}, u_{14}, u_{15}\}. \end{aligned} \tag{1}$$

Because each metric's importance is different, to determine the final evaluation outcome, we need to determine the weights of each metric in the ultimate result. In this study, we use the method of expert scoring. To avoid the cumulative effect of experts with the same type or similar background knowledge on the metric weights, we first clustered the experts through the cosine similarity between experts, and second, we calculated the different proportions of each expert by calculating the intra-class weights and interclass weights of the experts. They used the two methods of information entropy and clustering number proportion, respectively. Finally, the final metric weights are obtained based on the multiply sum of the initial values of the metrics given by each expert and the proportion of each expert. This method reflects the weight of individual experts and considers each expert's contribution to the metric weights.

Assume there are n experts scoring the importance of m metrics, and v_{ij} denotes the score of expert i scoring the importance of metric j, which finally constitutes the importance matrix V. According to the cosine similarity, we calculate the similarity between experts as follows:

$$V = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1m} \\ v_{21} & v_{22} & \cdots & v_{2m} \\ \vdots & \vdots & & \vdots \\ v_{n1} & v_{n1} & \cdots & v_{nm} \end{bmatrix}, \tag{2}$$

$$\begin{aligned} \text{sim} &= \frac{A \cdot B}{|A||B|} \\ &= \frac{\sum_{i=1}^{m} Ai \cdot Bi}{\sqrt{\sum_{i=1}^{m} (Ai)^2} \times \sqrt{\sum_{i=1}^{m} (Bi)^2}}. \end{aligned} \tag{3}$$

Assume e_i is the information entropy of the metric evaluation vector of expert i. Then,

$$e_i = -\frac{1}{\ln m} \sum_{j=1}^{m} f_{ij} \ln\left(f_{ij}\right), \quad i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m, \tag{4}$$

where $f_{ij} = V_{ij} / \sum_{j=1}^{m} V_{ij}$ is the weight of the importance of the j th metric in the i th expert rating vector to the sum of the evaluation of the m th metric.

Interclass expert weights: n experts are divided into l classes, and there are u_k experts in each class, and then, the weight of each class is given by:

$$\lambda_k = \frac{\mu_k^2}{\sum_{k=1}^{l} \mu_k^2}, \quad k = 1, 2, \ldots, l. \tag{5}$$

Intra-class expert weight: the entropy weight of the j th expert in the class is as follows:

$$\varphi_{ij} = \frac{e_i^{-1}}{\sum_{i=1}^{\mu_i} e_i^{-1}}. \tag{6}$$

Expert aggregate weights are defined by

$$\tau_i = \lambda_k \varphi_{ij}, \quad i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m. \tag{7}$$

Metric weights: after multiplying the weight vector of experts with the standardized importance matrix, the sum of columns of $\omega_{(ij)}$ is calculated; i.e., the terms of the same subscript are added to obtain a $1 * m$ vector, which is the metric weight vector and is given as follows:

$$\omega_{ij} = \tau_i f_{ij}, \quad i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m. \tag{8}$$

$$\sigma_j = \sum_{i=1}^{n} \omega_{ij}, \quad i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m. \tag{9}$$

Based on the expert similarity matrix, finally, we complete the clustering of experts. The specific Algorithm 1 is as follows.

3.2.2. Constructing a Tripartition. The core idea of the grey correlation analysis method is based on the similar program pairing between the various sequences in the entire system. The degree of association between the sequences is analyzed. The model requires only a small amount of samples for data analysis and has operational capabilities. It has the advantages of a simple operation method, convenient operation, and easy mining of data laws. Therefore, the grey correlation

Input: Initial scoring of metrics by experts V, n
Output: Weight of metrics W
(1) $*$ calculate expert similarity matrix $*$
(2) $S_{n\times n} \leftarrow \varnothing$
(3) for $i = 1$ to n do
(4) for $j = 2$ to n do
(5) $\text{sim}_{ij} \leftarrow \sum_{d=1}^{m} V_{id} \cdot V_{jd} / \sqrt{\sum_{i=1}^{m} (V_{id})^2} \times \sqrt{\sum_{i=1}^{m} (V_{jd})^2}$
(6) $*$ cluster similar experts $*$
(7) $Gr \leftarrow \varnothing, Dr \leftarrow \varnothing$ // Initialize the largest collection and class among experts.
(8) k = 0// Initialize the maximum number of similar classes among experts.
(9) if find $S_{\max} \leftarrow S$ and $S_{\max} \neq 0$ then
(10) $Gr \leftarrow S_{\max}$ corresponding two experts
(11) $\text{sim}_{AiBj} \leftarrow 0$
(12) $Dr \leftarrow Gr \cup S_{\max}$
(13) Repeat the above steps until $S = \varnothing$.
(14) $C \leftarrow \varnothing$ // Initialize the expert collection.
(15) for $i = 1$ to k-1 do
(16) for $j = 2$ to k do
(17) combine the collections containing the same experts in the pairwise clusters
(18) if unclustered experts then
(19) separate into a class
(20) $\lambda_k \leftarrow \mu_k^2 / \sum_{k=1}^{l} \mu_k^2$, $k = 1, 2, \ldots, l$.// the weight of each class
(21) $e_i \leftarrow -1/\ln m \sum_{j=1}^{m} f_{ij} \ln(f_{ij})$, $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$.// the metric evaluation vector of expert i
(22) $\varphi_{ij} \leftarrow e_i^{-1} / \sum_{i=1}^{\mu_i} e_i^{-1}$, $j = 1, 2..., l$.// the entropy weight of the j th expert in the class
(23) $\tau_i \leftarrow \lambda_k \varphi_{ij}$, $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$.// expert aggregate weights
(24) $f_{ij} \leftarrow V_{ij} / \sum_{j=1}^{m} V_{ij}$, $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$.
(25) $\omega_{ij} \leftarrow \tau_i f_{ij}, i = 1, 2, \ldots, n; j = 1, 2, \ldots, m$.
(26) $\sigma_j \leftarrow \sum_{i=1}^{n} \omega_{ij}, i = 1, 2, \ldots, n; j = 1, 2, \ldots, m$.// metric weights

Algorithm 1: Calculation of weights for expert clustering.

analysis model must perform simple operation analysis by extracting a small amount of sample data in a system. Then, the overall system can be analyzed. Development and change trends provide a quantitative measure. It is essentially a quantitative description of the dynamic development of the object methods of analysis and comparison. This method calculates the comparison sequence and reference that can reflect the behavior characteristics of the object. The degree of relevance between the sequences is used to sort and analyze the objects and finally get the results of the pros and cons of the objects.

The main steps of the traditional grey relational analysis model are as follows:

(1) Determining reference series and comparison series.
(2) Dimensionless processing of the sequence.
(3) Finding the grey correlation coefficient of reference series and comparison series.

Suppose a reference series is denoted as X_0, it has n comparison series, denoted as $X_1, X_2, ..., X_n$, and each comparison series X_i is associated with the reference series X_0 at various moments or under different behavior characteristics. The coefficient $V(X_0(k), X_i(k))$ can be calculated by the following formula:

$$V(X_0(k), X_i(k)) = \frac{\min_i \min_0 |X_0(k) - X_i(k)| + \xi \max_i \max_0 |X_0(k) - X_i(k)|}{\Delta_{i0}(l) + \xi \max_i \max_0 |X_{il} - X_{0l}|}, \tag{10}$$

where $\min_i \min_0 |X_{il} - X_{0l}|$ is the minimum difference in the second level, and $\max_i \max_0 |X_{il} - X_{0l}|$ is the maximum difference in the second level. The absolute difference is compared between each feature point on the sequence X_i and each feature point on the reference sequence X_0, and it is recorded as $|X_0(k) - X_i(k)|$. In general, the resolution coefficient ξ in the formula is generally 0.5.

(4) Finding the degree of grey relation $\gamma(X_0, X_1)$.

Each associated sequence and the selected reference sequence are all sequences composed of different moments or different characteristics. The correlation coefficient refers to the correlation degree value between the comparison sequence and the reference sequence at a particular time or feature. Usually, there are multiple values. There is a correlation

coefficient under each time or each feature. Because the information is too scattered, it is not conducive to the overall comparison of objects. Therefore, it is necessary to gather multiple correlation coefficients into one value. This value will be used as a quantitative representation of the degree of correlation between a comparison series and a reference series. Generally, the average value of the correlation coefficients at each time or feature is obtained. It indicates the degree of grey correlation, and the calculation formula is as follows:

$$\gamma(X_0, X_1) = \frac{1}{n}\sum_{k=1}^{n} V(X_0(k), X_i(k)), \tag{11}$$

where the value range of γ is $[0, 1]$. When the value of γ is closer to 1, the correlation between the two sequences is better, and the similarity is higher. The closer to 0, the opposite is true.

(5) Relevance ranking

Comparing the degree of association between different sequences is mainly by calculating the grey correlation value of n different comparison sequences to the same reference sequence and sorting them from largest to smallest, forming an association order, denoted as $\{x\}$, association. The sequence reflects the pros and cons of each comparison sequence. If $r_{0i} > r_{0j}$, it is said that $\{x_i\}$ is better than $\{x_j\}$ for the same reference sequence 0, which is recorded as $\{x_i\} > \{x_j\}$; r_{0i} represents the characteristic value of the i time comparison sequence to the reference sequence 0.

The study used the grey correlation analysis method to perform a comparative analysis of student sequences. The authors [30] dealt with quantifying qualitative indicators using an improved grey statistics-based approach. They combined the approximating ideal solution method with the grey correlation method to find out the weaknesses of teaching training and improve the assessment of training levels. The contribution of the work in the literature [31] solved the problem of weighting the evaluation indicators by weighing the different importance among the evaluation indicators through the correlation degree between the sequences. The work [32] also used this method to evaluate the weights of each evaluation index and, at the same time, combined with the theory related to the cloud model to complete the comprehensive evaluation of teaching quality. The literature [14] used the combination of grey correlation analysis and hierarchical analysis method to determine the weights of several factors. The grey correlation degree among each factor creatively established a hierarchical grey combination evaluation model and then judged the grade of internship teaching effect. This section utilizes the grey correlation analysis to construct the tripartition.

Assume there are n test samples, and the test result is $X_i' = (X_{i1}', X_{i2}', \ldots, X_{im}')$, $i = 1, 2, \ldots, n$, which represents the scores of these n test samples on m metrics.

Normalization of X_i' yields $X_i = (X_{i1}, X_{i2}, \ldots, X_{im})$, where X_{il} is the ratio of the component X_{il} to the mean of X_i' in the sequence X_i. That is,

$$X_{il} = \frac{X_{il}'}{\overline{X_i}}, \qquad \overline{X_i'} = \frac{1}{m}\sum_{l=1}^{m} X_{il}. \tag{12}$$

The optimal sequence is denoted as $X_0' = (X_{01}', X_{02}', \ldots, X_{0m}')$. In this study, the optimal value of each metric, i.e., the maximum value, is selected as the value of each component in the test data series and, after standardization, is noted as $X_0 = (X_{01}, X_{02}, \ldots, X_{0m})$.

The absolute value of the difference between X_i and X_0 at the l th component is noted as $|X_{il} - X_{0l}|$, and the minimum value of the difference between the comparison sequence i and the reference sequence 0 at m components is $\min|X_{il} - X_{0l}|$, $(l = 1, 2, \ldots, m)$, and the maximum value is $\max|X_{il} - X_{0l}|$, $(l = 1, 2, \ldots, m)$.

The absolute value of the difference between the n samples and the reference sequence is calculated separately, and the minimum value of all the differences is $\min_i \min_0 |X_{il} - X_{0l}|$, $(l = 1, 2, \ldots, m)$, abbreviated as $\Delta\min$, and the maximum value of all the differences is $\max_i \max_0 |X_{il} - X_{0l}|$, $(l = 1, 2, \ldots, m)$, abbreviated as $\Delta\max$. The formula to calculate the correlation coefficient between the sample sequence X_i and the comparison sequence X_0 is as follows:

$$\varepsilon_{i0} = \frac{\Delta\min + \rho\Delta\max}{\Delta_{i0}(l) + \rho\Delta\max} \quad (l = 1, 2, \ldots, m; i = 1, 2, \ldots, n) \rho \in [0, 1]. \tag{13}$$

From (11), it can be seen that the product of the discriminant coefficient ρ and $\Delta\max$ has a significant influence on the final result of the whole equation. The value of ρ impacts the overall contribution of $\Delta\max$ to the correlation degree. In general, ρ is taken as 0.5.

Based on equations (4) to (9), the metric weights can be calculated and denoted as $\sigma(l)$, $(l = 1, 2, \ldots, m)$. The weighted grey correlation between the comparison sequence i and the reference sequence 0 is denoted by r_{i0} and is calculated as follows:

$$r_{i0} = \sum_{l=1}^{m} \varepsilon_{i0}(l)\sigma(l). \tag{14}$$

The grey correlation values were between [0, 1]. Lager value means that the students' computational thinking skills are more similar to the optimal reference sequence, i.e., more excellent. To determine the percentage of students in each category, we can define two variables, a and b, and sort the students from largest to smallest based on the grey correlation. The top a% of students will be classified as excellent category, the bottom b% will be the passing category, and the rest will be medium.

3.3. An Illustrative Example. In this section, an example is given to verify the validity and reasonableness of the evaluation model. The experimental data are obtained from an online testing platform of a university. The data consist of two parts. The first part is the importance ratings of computational thinking indicators by six experts on a scale from 1 to 5, with higher values having the highest importance. The second part shows the test results of students in a school. Each student's test results for each metric were scored by 1–10. At the same time, a questionnaire was taken from the students. Moreover, the students self-evaluated their performance on each metric through self-awareness on a scale of 1–10. The final score matrix of the students was obtained as the mean of the scores of the two parts, teacher evaluation and self-evaluation.

First, the importance score matrix of the first-level metrics was given by six experts as follows:

$$\begin{bmatrix} 2 & 3 & 4 & 5 & 2 \\ 4 & 5 & 3 & 4 & 3 \\ 3 & 4 & 4 & 3 & 1 \\ 2 & 4 & 4 & 5 & 3 \\ 2 & 5 & 5 & 4 & 3 \end{bmatrix}. \tag{15}$$

The expert similarity matrix is calculated according to Equation 3.

$$\begin{bmatrix} 1. & 0.92487961 & 0.93771549 & 0.95734135 & 0.98872959 & 0.96025331 \\ 0.92487961 & 1. & 0.95397346 & 0.97989309 & 0.95229047 & 0.94837191 \\ 0.93771549 & 0.95397346 & 1. & 0.95398325 & 0.93724670 & 0.96101729 \\ 0.95734135 & 0.97989309 & 0.95398325 & 1. & 0.95714286 & 0.92786870 \\ 0.98872959 & 0.95229047 & 0.93724670 & 0.95714286 & 1. & 0.98165819 \\ 0.96025331 & 0.94837191 & 0.96101729 & 0.92786870 & 0.98165819 & 1 \end{bmatrix}. \tag{16}$$

The experts were clustered, and the clustering results were [1, 5, 6] for the first class, [2, 4] for the second class, and [3] for the third class. Next, the intra-class weights of the experts were calculated as follows: first class: 0.34081814, 0.32816024, and 0.33102162; second class: 0.49323828 and 0.50676172; and third class: 1. The weights between classes are as follows: 0.64285714, 0.28571429, and 0.07142857; the weights of 6 experts are as follows: 0.21909738, 0.14092522, 0.07142857, 0.14478906, 0.21096015, and 0.21279961.

According to equations (8) and (9), each expert weight is multiplied with the standardized metric importance vector, and the metric values with the same subscript are summed up. The final weight of the primary metric is obtained as [0.18465826, 0.23438986, 0.21356242, 0.2440862, 0.12330328].

Similarly, the secondary indicator weights can be calculated, and then, the primary and secondary indicator weights are combined to obtain the final secondary metric weights. The results are shown in Table 1. The meanings of some abbreviations in the table are as follows. FLM represents the first-level metrics, FLW represents the first-level weight, SLM represents the second-level metric, the ISLW represents the initial second-level weight, and FW represents the final weights.

A student's scores on each metric form a sequence that contains m scores, and a sample of n students forms an $n * m$ initial score matrix. Some of the student data are listed in Table 2.

In this study, the top 30% of students were selected as the excellent category, i.e., category A, the bottom 20% as the average category, i.e., category C, and the rest as the medium category, i.e., category B. Thus, the initial category classification of the evaluated subjects was completed. The initial classification of the three categories of students, that is, the tripartition, is $A = [10, 7, 24, 23, 20, 5, 6, 22, 18]$, $B = [9, 19, 4, 25, 13, 8, 21, 12, 28, 11, 15, 3, 17, 14, 29]$, and $C = [1, 26, 27, 16, 2, 0]$, respectively.

4. Association Rule Mining Based on Three-Way Classification

The evaluation system is gradually reformed, and the rating system has been steadily promoted. Compared with the refined scoring system, the rating is more conducive to promoting the progress and development of the evaluation objects. The two-branch classification will cause a more significant loss of misjudgment caused by the evaluation object. Multibranch classification divides the evaluation objects into excellent, good, medium, average, poor, or more fine-grained classification. This classification method increases a specific classification cost, and the teaching effect it brings is also open to question. The characteristics between categories are weakened, which is not conducive to mining. We are distinguishing features between categories. According to the characteristics of students' ability classification, this study introduces the three decision-making theories into the application of student ability classification, considering the relevance of evaluation objects, classifies students into three evaluation categories, and finally divides students into three categories: good, medium, and general. The correct classification of students can effectively reduce teaching costs and, at the same time, obtain more practical teaching effects.

Table 1: Metric weight table.

FLM	FLW	SLM	ISLW	FW
U_1	0.14935668	u_1	0.45571734	0.0681
		u_2	0.54428266	0.0813
U_2	0.23226924	u_3	0.28472944	0.0661
		u_4	0.33603961	0.0781
		u_5	0.37923096	0.0881
U_3	0.22308476	u_6	0.33742035	0.0753
		u_7	0.37019938	0.0826
		u_8	0.29238028	0.0652
U_4	0.25604126	u_9	0.14758813	0.0378
		u_{10}	0.27935669	0.0715
		u_{11}	0.35463976	0.0908
		u_{12}	0.21841543	0.0559
U_5	0.13924807	u_{13}	0.31673079	0.0441
		u_{14}	0.3267404	0.0455
		u_{15}	0.35652881	0.0496

Table 2: Student test data.

	u_1	u_2	u_3	u_4	u_5	...	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}
$S1$	7.75	8.00	7.75	6.25	7.00	...	6.75	5.50	6.25	8.25	4.25
$S2$	7.50	7.50	7.00	7.25	6.75	...	6.75	5.75	5.75	7.50	3.75
$S3$	8.75	7.50	7.25	6.25	7.00	...	7.25	6.25	5.75	8.50	5.25
$S4$	7.25	7.25	6.75	7.25	7.00	...	6.50	5.75	6.00	8.00	4.25
$S5$	7.75	7.50	6.75	7.25	7.25	...	7.25	6.00	6.75	8.00	4.75
$S6$	8.25	7.75	8.25	9.00	8.00	...	8.75	8.25	8.75	9.25	5.25
...						...					
$S25$	8.25	8.00	6.75	7.00	7.75	...	7.75	8.00	7.75	8.50	5.25
$S26$	7.75	7.75	7.00	8.25	6.00	...	7.50	7.25	8.00	8.25	6.00
$S27$	7.25	7.75	7.75	6.50	7.50	...	7.00	6.25	7.50	8.75	4.50
$S28$	6.50	7.50	7.25	6.25	6.50	...	6.00	5.50	5.50	6.75	4.00
$S29$	7.50	7.25	7.25	6.25	7.00	...	7.00	6.50	6.25	8.00	5.00
$S30$	8.25	8.50	7.50	6.50	7.75	...	7.00	7.50	7.25	9.00	5.25

The weighted correlation between the sequence of n student samples and the optimal reference sequence is [0.5156, 0.5976, 0.5751, 0.6097, 0.6759, 0.7066, 0.6903, 0.7265, 0.6657, 0.6889, 0.7278, 0.6296, 0.6581, 0.6641, 0.6085, 0.6227, 0.5727, 0.6078, 0.6911, 0.6736, 0.7187, 0.6603, 0.6889, 0.7216, 0.7262, 0.6673, 0.5907, 0.5819, 0.6404, 0.6066].

4.1. Three-Way Classification Based on Computational Thinking. The correlation between the metrics and the assessment results in the high-level areas, or in the middle-level areas, can be explored, which allows us to develop specific courses for students in the low-level areas and thus improve their computational thinking. This section will give the definition of three-way decision based on this example, the definition of three-way rule mining, and the specific example analysis process.

A three-way decision model with an ordered relationship is defined as follows.

Definition 1. Assume that OB is the set of students to be tested. E: $OB \longrightarrow (L, <)$ is an evaluation function on set OB. For $x \in OB$, $E(x)$ is an evaluation function value of x. Given a pair of thresholds $(\alpha, \beta) \in V \times V$ with $\beta \leq \alpha$, we divide OB into three pairwise disjoint regions:

$$\begin{aligned} P_1 &= \{x \in OB | e(x) \geq \alpha\}, \\ P_2 &= \{x \in OB | \beta < e(x) < \alpha\}, \\ P_3 &= \{x \in OB | e(x) \leq \beta\}. \end{aligned} \tag{17}$$

The three regions satisfy the following two conditions:

(1) $P_1 \cup P_2 \cup P_3 = OB$.

(2) $P_1 \cap P_2 = \Phi, P_1 \cap P_3 = \Phi, P_2 \cap P_3 = \Phi$.

According to the evaluation function $e(x)$, those objects greater than or equal to the value of the function are divided into a region P_1. Those objects less than or equal to the value of the function are divided into a region P_3, and objects in between are divided into a region P_2.

From the perspective of the TAO model, the division of the three regions allows us to better focus on each region and analyze each region's characteristics. We can identify those

metrics that can be improved, moreover develop some target strategies, and thus, we can improve the students' computational thinking. The direct outcome of the process is to move the students from relatively low-level regions to middle-level or high-level regions, that is, the movement-based three-way decision model, which was proposed by [22]. The movement-based three-way decision introduced actionable rules into the three-way decision, which means that a user can mine actionable rules and then produce the outcome of moving objects to generate benefits. The model aims to mine action strategy in three regions and move objects from unfavorable regions to favorable regions.

Definition 2. A decision table is a tuple as follows:

$$S = \left(OB, AT = A_s \cup A_h \cup A_f \cup \{d\}, \{V_a | a \in AT\}, \{I_a | a \in AT\}\right), \tag{18}$$

where OB is a nonempty finite set of objects, AT is a finite nonempty set consisting of attributes composed by three subsets, in which A_s is stable attributes, A_h is inert attributes that do not change easily but do change, A_f is flexible attributes, d is a decision attribute, V_a is a nonempty set of values for every attribute $a \in AT$, and I_a: $OB \longrightarrow V_a$ is a mapping. For every $x \in OB$, attribute $a \in AT$, and value $v \in V_a$, $I_a(x) = v$ means that the object x has the value v for attribute a.

Definition 3. Assume that $[x]$ and $[y]$ are equivalence classes in different regions. We can get two decision rules:

$$\begin{aligned} r_{[x]}&: \left[\bigwedge_{s \in A_s} s = f_s(x)\right] \wedge \left[\bigwedge_{f \in A_f} f = f_f(x)\right] \wedge \left[\bigwedge_{h \in A_h} f = f_h(x)\right] \Rightarrow d = f_d(x), \\ r_{[y]}&: \left[\bigwedge_{s \in A_s} s = f_s(y)\right] \wedge \left[\bigwedge_{f \in A_f} f = f_f(y)\right] \wedge \left[\bigwedge_{h \in A_h} f = f_h(x)\right] \Rightarrow d = f_d(y), \end{aligned} \tag{19}$$

where $r_{[\cdot]}, \cdot \in \{x, y\}$, is decision rule, A_s is a set of stable attributes, $f_s(\cdot)$ is the value of attribute s, A_f is a set of flexible attributes, A_h is inert attributes, $f_f(\cdot)$ is the value of attribute f, and $f_d(\cdot)$ is the value of decision attribute d.

Definition 4. Assume that $[x]$ and $[y]$ are equivalence classes in different regions, where $[x]$ is the equivalence class that is located in relatively low-level regions, such as low-level region and middle-level region, and the $[y]$ is the target equivalence class, which means that it is relatively high-level region. An ideal strategy is to make the $[x]$ equivalence class convertible to or close to convertible to the $[y]$ equivalence class; that is,

$$\begin{aligned} &r_{[x]} \uparrow r_{[y]}: \left[\bigwedge_{f \in A_f} f_f(x) \uparrow f_f(y)\right] \Rightarrow f_d(x) \uparrow f_d(y), \\ &\text{subject to} \left[\bigwedge_{s \in A_s} f_s(x) = f_s(y)\right] \wedge \left[\bigwedge_{h \in A_h} f_h(x) = f_h(y)\right], \end{aligned} \tag{20}$$

where $r_{[x]} \uparrow r_{[y]}$ is actionable rules from $[x]$ to $[y]$, $\bigwedge_{s \in A_s} f_s(x) = f_s(y)$ means that $[x]$ and $[y]$ have the same value of stable attributes, $\bigwedge_{h \in A_h} f_h(x) = f_h(y)$ means that $[x]$ and $[y]$ have the same value of inert attributes, and $\bigwedge_{h \in A_f} f_f(x) \uparrow f_f(y)$ means that the value of flexible attributes f is changed from $f_f(x)$ to $f_f(y)$.

The reason for introducing inert attributes is to strip away those attributes that do not change easily, even with much training under the teacher's strategic instruction, such as the student's IQ. These attributes may only change a little, even after prolonged training. They may be genetic in origin. Stripping these attributes may help teachers discover which characteristics are susceptible to instructional strategies.

In the following work, we analyze the association rules for the objects in the three regions. In particular, we use the Apriori algorithm to analyze students' computational thinking test data. In doing so, we can discover some strong association rule relationships among metrics and between metrics and assessment results in a large number of students' data. To reduce the cost of instructional strategy design, we divided these rules into three regions based on their frequency of occurrence: high-frequency rules, medium-frequency rules, and low-frequency rules. In other words, each area has three regions of rules. Then, teachers can choose specific teaching strategies to teach according to specific constraints, such as cost, so that students' computational thinking level can be improved and developed. This process is shown in Figure 3.

4.2. An Illustrative Example. By analyzing and mining the association rules that exist for each category of students and mining the association rules between indicators and test results, we can analyze the characteristics of students with different ability levels and can propose targeted improvement strategies to discover the characteristics of more capable students, thus having some positive significance for instructing weaker thinking students.

4.2.1. Test Data and Analysis. According to the test data, we assume that all student ratings made a collection $I = \{i_1, i_2, i_3, \ldots, i_k\}$ and the collection I consists of k different items, consisting of all indicators taken and the final

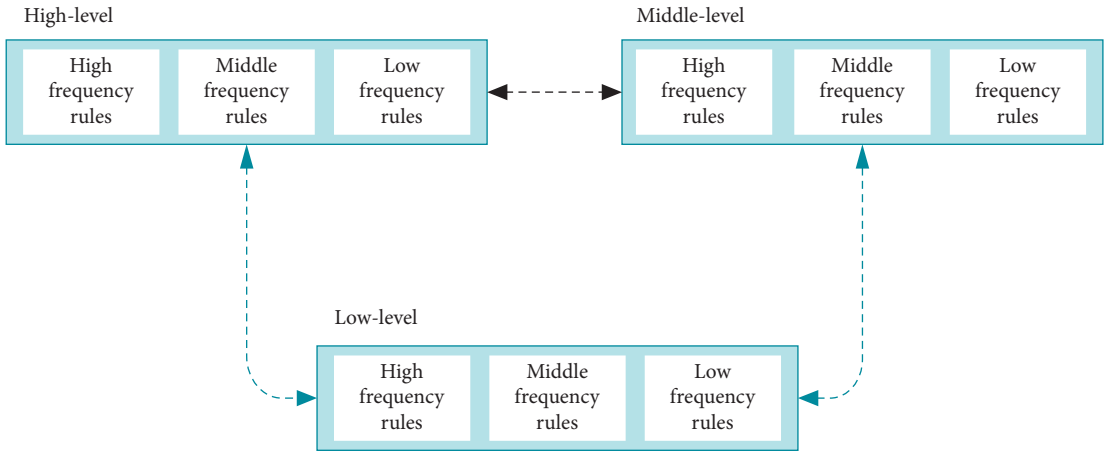

Figure 3: Rule mining based on movement-based three-way decision.

evaluation results. Each transaction T in the set $D = \{T_1, T_2, \ldots, T_q\}$ is a set of items in I. T_i is the set of scores of student i on each indicator and the final evaluation result.

The continuous data scores are discretized based on the student's score on each metric, to represent the continuous data scores, with 1–3 being a C, 4–6 being a B, and 7–10 being an A, resulting in all discrete grades, i.e., A, B, and C. We refer to all test data for each student as a transaction. Let us take the students of category I as an example and perform student feature mining.

The actual transaction data D are listed in Table 3. "TID" represents the test questions, and "Test Items" represents the grade on the five metrics and the final evaluation outcome.

The dataset D is scanned and the candidate set C1 is generated as shown in Table 4. The item in C1 with support less than the minimum support is removed, which in turn generates L1. Correspondingly, support level is as follows: "5B" is 0.889; "4A" is 0.889; "6A" is 0.778; "2B" is 0.778; and "3B" is 1, respectively. The set of items from the frequent item set is aggregated into the candidate set C2. Items in C2 are removed with support less than the minimum support, thus generating L2 as shown in Table 5. In a similar method, we can obtain C3 and L3, and C4 and L4 as shown in Tables 6–8, respectively. In this case, $K = 5$ is selected, and a total of 4 item sets are generated, as shown in Table 9.

4.2.2. Rule Mining. Through the analysis, we mined 63 strong association rules for the first category of students, that is, category A students. For example,

(1) "4A" ≥ "6A," confidence: 0.75

(2) "3B," "4A" ≥ "6A," confidence: 0.75

(3) "2B," "3B," "4A" ≥ "5B," confidence: 1.0

(4) . . .

The association rule shows that when the algorithmic ability is A, the level of computational thinking is A, and the pattern level has little effect on the computational thinking outcome. When the abstract level and pattern level are both B, the computational thinking outcome is only B even if the algorithmic test result is A. Therefore, when teachers instruct students, they should not only focus on students' algorithmic ability, but also focus on abstract understanding and pattern skill.

Similarly, we can analyze 16 strong association rules for intermediate students and a total of 54 strong association rules for average students. For example, the strong association rules for middle-level students include the following:

(1) "2B" ≥ "6B," confidence: 0.769

(2) "5B" ≥ "6B," confidence: 0.769

(3) "2B," "3B" ≥ "5B," confidence: 0.9

(4) . . .

The correlation rule shows that when the abstraction level is B, the corresponding computational thinking level is also B; when the evaluation level is B, the computational thinking level is also B, which indicates that both abstraction ability and evaluation ability influence computational thinking. In addition, the abstract level and pattern level have a grade of B, and the evaluation result is also B. Therefore, if we want to improve the evaluation ability, we should also improve the abstract and pattern ability accordingly.

For the average student, we found that:

(1) "2B," "4B," "3B" ≥ "6C," confidence: 0.799

(2) "3B" ≥ "4B," confidence: 1.0

(3) . . .

From the above rules, we find that when the levels of abstraction, pattern, and algorithm are all rated as B, the corresponding level of computational thinking is C. When the pattern is B, the level of the algorithm is also B. Therefore, there is a correlation between the level of pattern and the level of algorithm. Moreover, the level of the algorithm cannot be improved without the level of pattern, and the level of computational thinking cannot be improved without the level of abstraction, pattern, and algorithm.

By analyzing the association rules of students in different level areas, teachers can change the level of some metrics. That is, specific teaching methods and strategies are adopted

Table 3: Rule mining based on a three-way decision.

TID	Items
1	“1B,” “2B,” “3B,” “4B,” “5B,” “6A”
2	“1A,” “2B,” “3B,” “4A,” “5B,” “6A”
3	“1B,” “2B,” “3B,” “4A,” “5B,” “6A”
4	“1B,” “2B,” “3B,” “4A,” “5B,” “6A”
5	“1A,” “2A,” “3B,” “4A,” “5A,” “6A”
6	“1B,” “2B,” “3B,” “4A,” “5B,” “6A”
7	“1A,” “2A,” “3B,” “4A,” “5B,” “6A”
8	“1A,” “2B,” “3B,” “4A,” “5B,” “6B”
9	“1A,” “2B,” “3B,” “4A,” “5B,” “6B”

Table 4: Candidate set C1.

C1	Support level
1B	0.444
5B	0.889
1A	0.556
2B	0.778
5A	0.111
6B	0.222
6A	0.778
4B	0.111
2A	0.222
3B	1.0
4A	0.889

Table 5: Candidate set C2.

C2	Support level
“2B,” “6A”	0.556
“2B,” “3B”	0.778
“5B,” “2B”	0.778
“5B,” “3B”	0.889
“4A,” “5B”	0.778
“4A,” “6A”	0.667
“4A,” “3B”	0.889
“6A,” “3B”	0.778
“4A,” “2B”	0.667
“5B,” “6A”	0.667

Table 6: Frequent item set L2.

L2	Support level
“2B,” “3B”	0.778
“5B,” “2B”	0.778
“5B,” “3B”	0.889
“4A,” “5B”	0.778
“4A,” “6A”	0.667
“4A,” “3B”	0.889
“6A,” “3B”	0.778
“4A,” “2B”	0.667
“5B,” “6A”	0.667

Table 7: Candidate set C3.

C3	Support level
"4A," "5B," "2B"	0.667
"5B," "2B," "3B"	0.778
"4A," "5B," "3B"	0.778
"4A," "6A," "3B"	0.667
"4A," "5B," "6A"	0.556
"4A," "2B," "3B"	0.667
"5B," "6A,"" 3B"	0.778

Table 8: Frequent item set L3.

L3	Support level
"4A," "5B," "2B"	0.667
"5B," "2B," "3B"	0.778
"4A," "5B," "3B"	0.778
"4A," "6A," "3B"	0.667
"4A," "2B," "3B"	0.667
"5B," "6A," "3B"	0.778

Table 9: Candidate set C4 and frequent item set L4.

C4|L4	Support level
"4A," "5B," "2B," "3B"	0.667
"4A," "5B," "2B," "3B"	0.667

to change certain thinking skills, thus allowing students to transform from a lower level of computational thinking to a better level of computational thinking.

5. Conclusion

In this study, we explore research related to computational thinking, including strategies for developing it and means to measure it. We propose an evaluation model by combining grey correlation analysis, association rule, and three-way decision theory. The first step is to develop computational thinking evaluation indicators and then use a weighted grey correlation analysis-based approach to evaluate student's computational thinking skills. The weighted grey correlation between the student samples and the optimal reference sequence was considered, classifying tested students into three levels. Based on the initial classification results, the neighborhood of students was calculated based on the grey correlation between the evaluation objects, and each category of students was divided into positive, negative, and boundary domains, respectively.

We envision the future work to include, first, enriching and improving the evaluation indexes. Second, for the feature mining part after student classification, this study only applies the Apriori association rule mining algorithm. How to improve the rule mining also needs further research. Finally, the efficiency of this evaluation model and the system's performance also need to be improved. To analyze a large amount of student data, the classification method and the efficiency of feature mining need further research and exploration.

Data Availability

All data used during the study are available in a repository or online in accordance with funder data retention policies (https://archive.ics.uci.edu/ml/datasets.php, http://cs.uef.fi/sipu/datasets/).

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this study.

Acknowledgments

This work was supported in part by the Natural Science Foundation of Heilongjiang Province (LH2020F031) and Key Projects of Higher Education Reform in Heilongjiang Province (SJGZ20200084).

References

[1] J. M. Wing, "Computational thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33–35, 2006.

[2] C. Angeli and N. Valanides, "Developing young children's computational thinking with educational robotics: an interaction effect between gender and scaffolding strategy," *Computers in Human Behavior*, vol. 105, Article ID 105954, 2020.

[3] C. Chalmers, "Robotics and computational thinking in primary school," *International Journal of Child-Computer Interaction*, vol. 17, pp. 93–100, 2018.

[4] E. Relkin, L. E. de Ruiter, and M. U. Bers, "Learning to code and the acquisition of computational thinking by young children," *Computers & Education*, vol. 169, Article ID 104222, 2021.

[5] M. Özmutlu, D. Atay, and B. Erdoğan, "Collaboration and engagement based coding training to enhance children's computational thinking self-efficacy," *Thinking Skills and Creativity*, vol. 40, Article ID 100833, 2021.

[6] A. Gadzikowski, *Coding, Robotics, and Engineering for Young Students: A Tech Beginnings Curriculum (Grades Pre-K-2)*, Taylor & Francis, Hoboken, NJ, USA, 2021.

[7] J. R. Qu and P. K. Fok, "Cultivating students' computational thinking through student–robot interactions in robotics education," *International Journal of Technology and Design Education*, pp. 1–20, 2021.

[8] M. Chevalier, C. Giang, A. Piatti, and F. Mondada, "Fostering computational thinking through educational robotics: a model for creative computational problem solving," *International Journal of STEM Education*, vol. 7, no. 1, pp. 1–18, 2020.

[9] M. Xiao and X. Yu, "A model of cultivating computational thinking based on visual programming," *IEEE*, in *Proceedings of the 2017 International Conference of Educational Innovation through Technology (EITT)*, pp. 75–80, Osaka, Japan, December 2017.

[10] P. Vesikivi, M. Lakkala, J. Holvikivi, and H. Muukkonen, "The impact of project-based learning curriculum on first-year retention, study experiences, and knowledge work competence," *Research Papers in Education*, vol. 35, no. 1, pp. 64–81, 2020.

[11] Z. Cui and O. L. Ng, "The interplay between mathematical and computational thinking in primary school student's mathematical problem-solving within a programming environment," *Journal of Educational Computing Research*, vol. 59, no. 5, pp. 988–1012, 2021.

[12] S. Grover, S. Basu, M. Bienkowski, M. Eagle, N. Diana, and J. Stamper, "A framework for using hypothesis-driven approaches to support data-driven learning analytics in measuring computational thinking in block-based programming environments," *ACM Transactions on Computing Education*, vol. 17, no. 3, pp. 1–25, 2017.

[13] J. H. Ku, "Designing an app inventor curriculum for computational thinking based non-majors software education," *Journal of Convergence for Information Technology*, vol. 7, no. 1, pp. 61–66, 2017.

[14] B. Zhong, Q. Wang, J. Chen, and Y. Li, "An exploration of three-dimensional integrated assessment for computational thinking," *Journal of Educational Computing Research*, vol. 53, no. 4, pp. 562–590, 2016.

[15] N. L. Fanchamps, L. Slangen, M. Specht, and P. Hennissen, "The impact of SRA-programming on computational thinking in a visual oriented programming environment," *Education and Information Technologies*, vol. 26, no. 5, pp. 6479–6498, 2021.

[16] M. Román-González, J. C. Pérez-González, and C. Jiménez-Fernández, "Which cognitive abilities underlie computational thinking? criterion validity of the computational thinking test," *Computers in Human Behavior*, vol. 72, pp. 678–691, 2017.

[17] Ö. Korkmaz, R. Çakir, and M. Y. Özden, "A validity and reliability study of the computational thinking scales (cts)," *Computers in Human Behavior*, vol. 72, pp. 558–569, 2017.

[18] S. C. Kong, M. M. Chiu, and M. Lai, "A study of primary school student's interest, collaboration attitude, and programming empowerment in computational thinking education," *Computers & Education*, vol. 127, pp. 178–189, 2018.

[19] M. J. Tsai, J. C. Liang, and C. Y. Hsu, "The computational thinking scale for computer literacy education," *Journal of Educational Computing Research*, vol. 59, no. 4, pp. 579–602, 2021.

[20] R. Ali, S. Lee, and T. C. Chung, "Accurate multi-criteria decision making methodology for recommending machine learning algorithm," *Expert Systems with Applications*, vol. 71, pp. 257–278, 2017.

[21] R. Ali, M. Afzal, M. Sadiq et al., "Knowledge-based reasoning and recommendation framework for intelligent decision making," *Expert Systems*, vol. 35, no. 2, Article ID e12242, 2018.

[22] Y. Yao, "An outline of a theory of three-way decisions," in *Proceedings of the International Conference on Rough Sets and Current Trends in Computing*, vol. 695, pp. 1–17, Springer, Madrid, Spain, July 2012.

[23] J. Xu, Y. Zhang, and D. Miao, "Three-way confusion matrix for classification: a measure driven view," *Information Sciences*, vol. 507, pp. 772–794, 2020.

[24] C. Jiang, D. Guo, Y. Duan, and Y. Liu, "Strategy selection under entropy measures in movement-based three-way decision," *International Journal of Approximate Reasoning*, vol. 119, pp. 280–291, 2020.

[25] C. Jiang, D. Guo, and R. Xu, "Measuring the outcome of movement-based three-way decision using proportional utility functions," *Applied Intelligence*, vol. 51, pp. 1–15, 2021.

[26] C. Jiang, D. Guo, and L. Sun, "Effectiveness measure for TAO model of three-way decisions with interval set," *Journal of Intelligent and Fuzzy Systems*, vol. 40, no. 6, pp. 11071–11084, 2021.

[27] A. Fluck, M. Webb, M. Cox et al., "Arguing for computer science in the school curriculum," *Journal of educational technology & society*, vol. 19, no. 3, pp. 38–46, 2016.

[28] K. Brennan and M. Resnick, "New frameworks for studying and assessing the development of computational thinking," in *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, p. 25, Vancouver, Canada, April 2012.

[29] C. Selby and J. Woollard, "Computational thinking: the developing definition," in *Proceedings of the Special Interest Group on Computer Science Education (SIGCSE)*, Atlanta, GA, USA, March 2013.

[30] S. S. Mahapatra and B. N. Panda, "Benchmarking of rapid prototyping systems using grey relational analysis," *International Journal of Services and Operations Management*, vol. 16, no. 4, pp. 460–477, 2013.

[31] X. Xu, Y. Wang, and S. Yu, "Teaching performance evaluation in smart campus," *IEEE Access*, vol. 6, pp. 77754–77766, 2018.

[32] L. Y. Zhai, L. P. Khoo, and Z. W. Zhong, "Design concept evaluation in product development using rough sets and grey relation analysis," *Expert Systems with Applications*, vol. 36, no. 3, pp. 7072–7079, 2009.