International
Journal
Of Advanced Research in Engineering & Management (IJAREM)
ISSN: 2456-2033 || PP. 01-04

Exploration of Discrete Mathematics Teaching Models from the Perspective of Computational Thinking

Jing Kong[1,*], Liang Fang[2], Rui Chen[3]
College of Mathematics and Statistics, Taishan University, Tai'an, Shandong, China

Abstract: Computational thinking is an analytical idea and an important educational idea. Discrete mathematics is a compulsory course for computer science majors, which contains a lot of abstract thinking in the content of the course, and plays an important role in the development of students' logical thinking ability and comprehensive quality and so on. This paper mainly analyses the problems existing in the teaching of discrete mathematics from the perspective of practical teaching, and discusses how to intersperse computational thinking in the teaching process of discrete mathematics through the clever setting of cases.
Keywords: Discrete mathematics; Computational thinking; Teaching methods; classroom instruction

I. INTRODUCTION

In recent years, with the rapid development of information technology, computer technology has penetrated into physics, finance, biology, medicine, business, manufacturing and securities and other fields and industries, and with these disciplines, the cross between the industry is more and more in-depth, has become an integral part of them. The rapid development of computer technology has completely affected people's life, learning and work, and similarly, the use of new products generated by information technology has also affected people's way of thinking and thinking ability.

The concept of computational thinking was first introduced by Prof. Yizhen Zhou of Carnegie Mellon University in Communications of the ACM in 2006 [1]. Later, Cuny, Snyder and Wing gave the definition that computational thinking refers to the thought process of interpreting a problem and its solution and then representing the solution into a form that can be effectively realized by an information processing agent. It can be seen that the essence of computational thinking is abstraction and automation; from the connotation point of view, computational thinking is how human beings think like computers; from the extension point of view, computational thinking is the use of computer science concepts related to human behavioral understanding, problem solving, system design, as well as a series of thinking activities related to computer science . The concept of computational thinking has attracted a lot of attention in the field of computer education after it was proposed, and computational thinking is regarded as a way of thinking that all people must have, just like reading, writing, and doing arithmetic, and it has become a basic skill that college students must have in the 21st century.In 2010, the Teaching Guidance Committee of Computer Science and Technology for Higher Education of the Ministry of Education clearly pointed out in the "Composition and Cultivation of Professional Abilities of Computer Science and Technology Specialists in Higher Education" that In 2010, the Steering Committee for Teaching Computer Science and Technology in Higher Education of the Ministry of Education in "Composition and Cultivation of Computer Science and Technology Professionals in Higher Educational Institutions" explicitly pointed out that computational thinking should be an important part of computer science teaching. At present, many scholars have explored how to cultivate students' computational thinking ability in the professional courses of computer science majors [2-5].

II. THE TEACHING OF DISCRETE MATHEMATICS AND THE DEVELOPMENT OF COMPUTATIONAL THINKING

Discrete mathematics is a branch of modern mathematics, and its main research object is the structure of discrete quantities and the interrelationship between quantities of a science [6]. As a basic course for computer-related majors, discrete mathematics contains very rich content, and the teaching content covers a number of mathematical branches such as mathematical logic, set theory, algebraic systems (algebraic structures), graph theory foundations and number theory foundations. Discrete Mathematics covers these branches of

mathematics in a self-contained system, while at the same time, each branch is more or less related to each other. The content of the course is full of definitions, symbols, formulas, theorems, theorems and proofs, highly theoretical and highly abstract. Students in the learning process generally believe that the course is a difficult course. The main reasons are as follows.

(1) Discrete mathematics consists of four main parts: mathematical logic, set theory, algebraic structure, graph theory. These four parts are independent of each other and form their own system. It is difficult for students to find the connection between them in the process of learning, and thus the purpose of learning the course is not clear.

(2) An important feature of discrete mathematics is formalization and symbolization. This formal model contains a large number of letters, symbols, formulas, graphs and so on. In the teaching process, the course has fewer hours, and many definitions, theorems and proofs have to be taught in class, so students cannot understand these abstract concepts in a short time, and they can only memorize them and fail to recognize the substantive meaning of these abstract models.

(3) Due to the large number of formal models and strong theory in the course content, students tend to study it as a mathematical course, not clear about the connection between it and computer science, and do not know enough about the status and role of this course in the computer science profession. Most of the contents do not have experimental links, students can not see the practical application of these formal models, so they can not appreciate the specific application of the course in computer science, lack of interest in learning.

(4) In the actual teaching process, most teachers teach according to the traditional teaching methods, first introduce the symbols, definitions, theorems, proofs and other basic theories, and then use examples to deepen students' understanding of the basic knowledge, and finally set up after-school homework for extracurricular reinforcement. However, this teaching method has great disadvantages for cultivating students' computational thinking ability, and it does not combine with computer specialization to focus on the application ability training, so that students can't link the theoretical knowledge and the practical application of computers in the learning process.

Computational thinking offers a fresh approach to teaching and learning. Although the contents in discrete mathematics are independent, there is a certain connection between these contents when analyzed from an abstract point of view. At the same time, the teaching purpose of these contents is to train students' abstract thinking ability, so that students can use discrete structure to establish an abstract model of the actual problem and build an algorithm for solving the problem on the basis of this, which reflects the two core contents of abstraction and automation in computational thinking. Therefore, the training of computational thinking ability should be strengthened in the discrete mathematics course, and the teaching content of the course should be organized from the perspective of computational thinking in the teaching process. The following is a description of the method of teaching with computational thinking on several discrete mathematics problems.

III. PRACTICE OF TEACHING DISCRETE MATHEMATICS BASED ON THE CULTIVATION OF COMPUTATIONAL THINKING

Case 1: Computational Thinking Skills Development in Teaching Equivalence Relationships

Equivalence is a very important relationship, and the properties of highly abstract equivalence relationships are: self-reversal, symmetry, and transitivity. It is difficult for students to understand, especially its applications. The most important application of equivalence relations is the division of sets. What use is division to computers? Explaining it here in terms of computational thinking will make students understand it quickly and stimulate their research interest. Computer classification is actually a computer recognition problem, talking about the computer recognition problem, the current application of more, for example, highway toll booths, automatic license plate recognition, door into the system of fingerprint identification, etc., how to identify the computer? This needs to be clear about its principles. After the computer collects information to rely on the results of the calculation to identify, in a large number of objects, to identify an object, the object is not every time the collection of data is accurate and consistent, these are not consistent with the data pointing to the same object, should they be the same equivalence class? Obviously should be. How to ensure their categorization? This is to seek a reasonable equivalence relationship, thus, the abstract equivalence relationship problem becomes a practical application problem, in the face of a specific classification (identification) problem, is it to find a specific equivalence relationship? If this equivalence relation is found, the algorithm is found, and the computer can be allowed to help

the human to accomplish this work. Thus, the equivalence relation is a separator with which to separate out the various types of things that are mixed together. From such a point of view to analyze, explain, students not only will not resist its abstraction, but also full of enthusiasm to study, the teaching effect is naturally very good.

Case 2: The development of computational thinking skills in teaching prefix code problems

Discrete Mathematics will talk about coding when introducing Havermann Tree applications, and there is a requirement for coding that a code cannot be a prefix code to any other code. Some students don't understand why this is required. Why wouldn't it work without such a restriction, for example by using the delimiter approach? The problem can be described with an example. If there are two codes, one is "110" (for character A), the other is "110110" (for character B), according to the coding rules, this is a non-compliant code, because "110" is "110110". "is the prefix of "110110". However, if a delimiter symbol "'" to deal with, the string becomes "110'110110", in the search for reading, found that the delimiter symbol when the output character, or The characters "A" and "B" can be output correctly. To explain this problem, but also with the basic concept of computational thinking, computational thinking, another basic principle is that computational activities must be adapted to the structural characteristics of the computer itself. In accordance with the computer storage and processing structure, the data stored in memory, to "page" as a unit read to the CPU processing, assuming that now read to the CPU in the page data, the last 3 bits of data is "110", then is the output directly "A" directly? Or should we wait until the next page of data comes in? According to the rule of delimiters, it must be waiting for the next page of data to enter before deciding whether or not to translate the current. This reduces the efficiency of the computer. In addition, non-equal-length encoding itself is to compress the total length of the decoding problem, plus the delimiter increases the total length of the decoding, which is inconsistent with the original intent. To summarize, one is for efficiency and the other is for the goal.

Case 3: The development of computational thinking skills in the teaching of Euler and Hamilton graphs

When teaching Euler's and Hamilton's graphs, one can start with the origin of graph theory, the Seven Bridges Problem of Königsberg. Euler then studied the seven bridges problem when the shape, length, width and other factors of the bridge are discarded, with a side to represent the bridge, with a point to represent the location connected to the bridge, so as to construct an undirected graph model. Then, two decision theorems were summarized on the basis of telling how Euler thought at that time. Finally, the students can deeply understand how to determine whether there is Euler's graph and Euler's path in a graph by playing a one-stroke game. The same is true for Hamiltonian graphs, starting from the problem of traveling around the world, guiding students to abstract the essence of the problem through thinking, and then leading to the necessary and sufficient conditions for determining Hamiltonian graphs. Word Solitaire is a good example of a generalization of these two points of knowledge by comparing the differences between the definition and the conditions for determining them. The rule is to assume that there are multiple cards, each of which has an English word written on it, and that the cards now have to be connected into a string in a certain order. It is required that the first letter of the word on each card (the first card may be excepted) be exactly the last letter of the word on the previous card, and that each card be used only once (case-insensitive). There are two abstract models that students might give for this problem. The first approach is natural, treating each word as a vertex, with an edge between the two words when and only when the last letter of the previous word is the same as the first letter of the word that follows. In this way the word solitaire problem is transformed into a problem of finding a Hamiltonian pathway. The second method is to consider all the letters of the alphabet as 26 nodes, each word as a directed edge, the beginning point of the directed edge is the first letter of the word, the end point is the last letter of the word. In this way, the word solitaire problem is transformed into a problem of finding Euler's path. Comparing these two methods, the first method is NP-hard problem and the second method has polynomial time complexity. Clearly the second is the better method. Through this example, students were guided to abstract the concrete problems and compare the computability and complexity of various solutions, thus effectively training their computational thinking.

IV. CONCLUSIONS

Computational thinking is a form of thinking that is consistent with the working activities of computers. Accurately grasping and utilizing computational thinking in teaching can more smoothly explain to students the origin of some problems and the rationality of their methodological choices. For students with low computer

literacy, using computational thinking to help them understand computer knowledge correctly is not only easy to understand, but also can stimulate their interest in studying computer knowledge.

Combining computational thinking in the teaching of discrete mathematics requires teachers to continuously explore and summarize their experiences in teaching practice, present the essence of the course to students, and effectively improve students' abilities in logical thinking, computational thinking and innovative thinking. We will follow up with the basis of the existing teaching reform of discrete mathematics courses to further explore and study in depth the innovation of teaching mode to cultivate students' computational thinking ability.

REFERENCES

[1] J. M.Wing, Computational Thinking, *Communications of the ACM*，49（3）2006，33-35.
[2] C. Liang, X. Zhoubo, G. Tianlong et al. Cultivation of Computational Thinking in Discrete Mathematics Teaching, *Computer Educatio*n, 7(14) 2011:90-94.
[3] L. Jingming, H. Xiande, L. Yun, Cheng Jiaxing, Research on the design of discrete mathematics classroom teaching program based on computational thinking, *Journal of Pu'er College*, 31(3), 2015, 124-126.
[4] Z. Ruixia, Z. Fan, Y. Guozeng, An investigation on the tiered teaching of discrete mathematics based on the cultivation of computational thinking ability, *Zhengzhou Normal Education*, 10(6), 2021,70-72.
[5] F. Liu, Computational Thinking Cultivation in Propositional Logic in Discrete Mathematics Courses, *Computer Education*, 11 (04), 2021, 151-154.
[6] G. S. Yun, Q. W. Ling, Z. L. Ang, *Discrete mathematics* (Tsinghua University Press, Beijing, 2021).