Shih, JL. et al. (Eds.) (2023). Proceedings of the 31st International Conference on Computers in Education.
Asia-Pacific Society for Computers in Education

Analysis of algorithmic strategy development in the development of computational thinking of upper elementary school students

Xiaowen WANG, Pinqi HU & Guang CHEN*
Faculty of Education, Beijing Normal University, Beijing, China
*guang@bun.edu.cn

Abstract: Computational thinking has become a thinking ability that we need to master in our daily lives. Programming is a way to cultivate computational thinking. This study focuses on the algorithmic links of senior primary school students in programming activities, puts forward a hierarchical table of algorithmic strategies, and analyzes their strategic development paths according to their programming behaviors.In this paper, Swift Playgrounds, developed by Apple Company, is selected as the tool of programming activities, and three levels are selected as the teaching content in the three theme activities of Learning Programming 1 in Swift Playgrounds software. The target audience of the activities is the senior (fourth to sixth grade) students in primary schools. This paper studies the design of the micro-generation method, carries out programming activities based on computational thinking for students through pre-test, practice, post-test and migration stages, and collects data by interview and classroom observation so as to analyze the development path of students' algorithm strategies in different stages and the change track of algorithm strategies in each stage.This study provides some support for teachers' teaching in programming teaching activities.

Keywords: Programming teaching activities, computational thinking, micro-genetic method, algorithm strategy

1. Introduction

In the context of the rapid development of information technology, society has put forward new requirements for people's abilities and accomplishments. People should learn to understand some problems encountered in social production, life and learning from the perspective of computer science, and be able to use the basic principles of computer science and advanced technical tools to solve complex and difficult problems (Tedre & Denning, 2016).

Computational thinking is a key research topic in computer education. Different researchers have different definitions of computational thinking, but the definition of computational thinking proposed by Jeannette, which includes five elements: algorithms, decomposition, abstraction, generalization and debugging, has been widely recognized by researchers all over the world. Jeannette believes that computational thinking includes thinking activities such as problem-solving, designing systems, and understanding human behavior. This is not only a skill that scientists need to possess, but also a skill that every one of us needs to possess (Jeannette, 2006).

As one of the ways to cultivate computational thinking, programming education can cultivate students' systematic problem-solving and strategic thinking abilities. In the process of programming, students can not only learn some basic knowledge but also learn logical methods and problem-solving strategies through specific problem situations, so it becomes a powerful starting point for cultivating computational thinking (Winslow et al. ,2018).

According to the cognitive development level and thinking development process of primary school students, primary school students above grade three are gradually transitioning from image generalization level to abstract generalization level, and their reasoning ability is also in a stage of rapid development. They can independently and logically demonstrate more complicated judgments and processes. Therefore, at this stage, students' computational thinking can be cultivated through programming teaching. For example, Saez-Lopez et al.(2016) through on-the-spot evaluation of the effect of using Scratch's visual programming language in classroom practice, explore whether students have made significant progress in learning programming concepts, logic and computational practice; Zhao & Shute(2019) asked students to play "Penguin Go" video game in the classroom, that is, to create a program to guide penguins to cultivate students' cognition and attitude towards the development of computational thinking skills through a regional approach; On the other hand, Chen-Chung et al.(2011) designed a simulated game to let students learn and use programming knowledge to solve situational problems, and explored the influence of simulated games on learning experience and problem-solving strategies related to programming knowledge. Pellas & Peroutseas(2016) showed students their participation in a 3D multi-user game environment by combining Second Life with Scratch4SL's 2D programming environment and used them to develop computational thinking skills in collaborative problem-based programming tasks.

To sum up, most of the current research on computational thinking focuses on the development and effect research of computational thinking activity courses, and the research on students' behavior in the process of computational thinking is relatively lacking. This study focuses on the cultivation of students' algorithm strategies in computational thinking activities. Starting with the basic programming knowledge, it studies the strategies and behaviors used by students when using programming knowledge to solve problems, and provides support for front-line teachers' programming teaching process.

Therefore, this study intends to solve the following problems:

(1) What are the algorithm strategies that students mainly use at different stages of different programming activities?

(2) What are the development paths of students' algorithms and strategies at different stages of the same programming activity?

2. Methods

In this study, the micro-generation method is used to study the changing process of students' algorithm strategies in the programming process, which is an exploratory preliminary study. This study mainly analyzes the development of algorithm strategy among senior primary school students in Swift Playgrounds-based programming activities, and further analyzes the path of students' strategy development according to the algorithm strategy grade table.

The whole teaching process of programming activities is divided into three thematic activities: command, function and For loop. Each thematic activity is divided into four stages: pre-test, exercise and post-test. After the completion of the three thematic activities, a complete programming project is carried out and used as a test in the migration stage. And through interviews and classroom observation to collect data. The interview method is mainly used to interview students after each stage in the process of teaching by the micro-generation method so that students can orally express what strategies are used to complete this part of the challenge and why such strategies are used to solve the problem. Classroom observation is mainly used to track and record the development of students' algorithm strategies in class so as to further analyze students' algorithm strategies.

The research object of this study is the students in senior grades (grade four to grade six) in primary schools, and they are recruited through online registration. A total of 15 students were recruited, including 12 boys and 3 girls, aged between 10 and 12, with an average age of 10.9 years and a standard deviation of 0.59. There are 12 fourth-grade students, 2 fifth-grade students and 1 sixth-grade student. During the implementation of the activity, a total of 15 students participated, but 2 students failed to complete all the courses. After the

implementation of the activity, a total of 15 students' learning data were received, of which 13 were valid. Among the valid data collected, from the point of view of school distribution, 9 people are studying in primary schools in Beijing, while 4 people finish their usual studies at home. In terms of gender, there are 10 boys and 3 girls, with an average age of 10.2 years and a standard deviation of 0.38.

This experiment is divided into three courses, and the content taught in each course is exactly the same. During the class, each student is recorded on an iPod Touch. At the same time, the iPad used by the students turns on the screen recording function, and a corresponding number of graduate teaching assistants are equipped for stage interviews and behavior tracking.

In this experiment, there are 15 subjects, who have completed 12 levels of programming game challenges in 3 theme activities, and designed 19 interview questions in the pre-test, practice, post-test and migration stages. Based on this, the obtained data is encoded. Combined with the specific programming behavior of middle school students in this research course, the algorithm strategies used by students in this programming process are summarized and classified, and according to the path and order of children's cognitive development from low-level strategies to high-level strategies, three experts in the field of psychology are invited to divide students' algorithm strategies into four levels, namely, zero-level strategies, first-level strategies, second-level strategies and third-level strategies. According to the different development characteristics of each level strategy, combined with students' specific algorithm behavior, the different level strategy characteristics of algorithm strategy are defined, and according to the video analysis and interview answers of 13 subjects' programming activities in Study 2, three experts teach coders to encode independently, finally improve the algorithm strategy level, and analyze the reliability according to the consistency percentage formula proposed by Holsti in content analysis. The r (reliability) of the algorithm strategy rating table is 0.9, and the reliability of the evaluation records of three judges, A, B and C, is greater than 0.9, which shows that the evaluation result of the main judge can be used as the result of content analysis, which achieves the consistency of coders. The strategy level rating table is shown in the following table:

Table 1. *Level Table of Algorithm Strategy*

Algorithm Strategy Level	Definition
Level 0	In the process of the algorithm, we rely on our own subjective judgment and experience to complete the algorithm.
Level 1	On the basis of subjective judgment, simple methods are used to complete the level goal, and the simple methods are limited to the commands involved in the level goal.
Level 2	On the basis of simple methods, use appropriate methods to complete the level goal, but it is limited to simpler scene goals.
Level 3	On the basis of appropriate methods, draw inferences from others and think independently about the best method needed to complete the checkpoint goal.

3. Results and Discussion

This chapter will analyze the data collected in the previous section. The study mainly uses qualitative analysis to investigate the changes in students' strategies and behaviors in the process of programming. It mainly uses encoding the video of students' entry into the classroom, and analyzing the interview contents in the pre-test, practice, post-test and migration stages to discuss the changes in students' strategies in the programming process. According to the micro-generation method, the breadth analysis of the same programming activity in pre-test, practice, post-test and migration stages is carried out. Mainly through the three theme activities, students' individual development routes are carried out, and the individual development roadmap is drawn according to the development of children's strategic

level at different stages. Based on this road map, this paper analyzes the migration degree of students' algorithm strategies of three theme activities in the migration stage.

3.1 The development route of individual algorithm strategy in command theme activities.

From the roadmap of individual development of the algorithm, it can be seen that five people in the algorithm strategy of command theme activity have developed from Level 0 to Level 2; One person developed from Level 0 to Level 2 and then dropped to Level 0 and then to Level 2; Two people developed from Level 0 to Level 2 and then fell to Level 1 and then to Level 2; Five people directly developed from Level 1 to Level 2. There are four strategic development paths in total.

From the development path of algorithm strategy, all students use low-level strategy in the pre-test stage of command theme activities. In the practice stage, students can easily master this method after teachers teach the method of counting grids and planning the shortest route, so the second-level strategy is used the most frequently in the post-test stage, but a small number of students still use low-level strategy. In the migration stage, students' algorithms all develop into high-level strategies, which also shows that in the algorithm strategy of command sentences, low-level strategy is a monotonically decreasing development model, and high-level strategy.

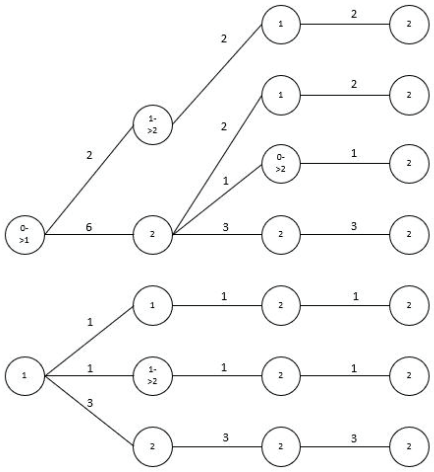

Figure 1. Roadmap of individual development of command theme activity algorithm strategy

3.2 The development route of individual algorithm strategy in function theme activities.

From the roadmap of individual development of the algorithm, it can be seen that four people in the algorithm strategy of function theme activity have developed from Level 1 to Level 3; Five people have developed from Level 1 to Level 3 and then dropped to Level 1; Four people developed from Level 1 to Level 2 and then fell to Level 1. There are three strategic development paths in total.

In the development path of algorithm strategy, all students use low-level strategy in the pre-test stage. Through the function method teaching in the practice stage, students' algorithm strategy has reached the second and third levels, but in the migration stage, Level 2 has dropped to Level 0, the most frequently used strategy level is the first-level strategy, and the third-level strategy is also used by students. It shows that in this part of the function, students do not fully grasp the definition and usage of the function and learn to draw inferences from others. The usage times of low-level strategies and high-level strategies are evenly distributed, and the students' algorithm strategies are not developing from low-level strategies to high-level strategies, but the development modes of low-level strategies and high-level strategies fluctuate and alternate constantly.

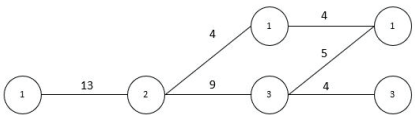

Figure 2. Roadmap of individual development of function theme activity algorithm strategy

3.3 The development route of the overall algorithm strategy for loop theme activity.

From the roadmap of individual development of the algorithm, it can be seen that one person in the algorithm strategy of “For loop” theme activity has developed from Level 1 to Level 2; There are 12 people who have always maintained the level of secondary strategy. There are two strategic development paths in total.

In the theme activity of “For loop”, students' algorithm and strategy have maintained a high level from the pre-test stage to the migration stage, even though the most frequently used strategy in the post-test stage is the first-level strategy. This is because in the post-test stage, it is necessary to customize the function and use the loop to write the code. According to the development map of algorithm and strategy in the function part, students can master the definition and usage of For loop skillfully in this part. So on the whole, the development of advanced strategies is constant.

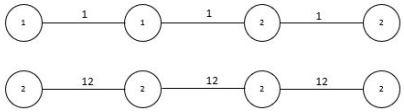

Figure 3. Roadmap of Individual Development of Algorithm Strategy for Cyclic Theme Activity

Synthesizing the development paths of algorithm strategies of the three theme activities, we can get the following results: the proportion of algorithm strategies used by middle school students in command activities and “For loop” activities is getting higher and higher, and the overall trend is from low-level strategies to high-level strategies, but in the later period, low-level strategies have not all disappeared, and some students still use low-level strategies to solve problems; The strategy development path of the function part shows this point more clearly. The algorithmic strategy of the function part develops alternately. Even if students learn more advanced function knowledge in the practice stage, more students still choose lower-level codes to replace the function strategy in the migration stage, which shows that students' cognitive strategies do not completely develop from low-level strategies to advanced strategies, but students subconsciously choose their most skilled and direct strategies by relying on their own knowledge and pre-concept experience in different situations.

4. Conclusion

This study uses the micro-generation method to study the changes of students' strategic behavior in programming activities, and makes a preliminary exploration in programming education. Through this study, it was found that students use different algorithm strategies when completing different programming activities. In such simple programming activities as command statements and For loop, students can use advanced strategies taught by teachers to complete tasks, which shows that students' algorithm strategies are constantly developing from low level to high level, and the migration effect of algorithm strategies is obvious. However, in such difficult programming activities as function statements, the algorithm strategies used by students before and after teacher intervention are not the methods taught by teachers, but their own subconscious strategies, which shows that the algorithm strategies have not developed from low level to high level in this link, and the transfer effect is not obvious. Therefore, the development of cognitive strategies is complicated, and students need certain cognitive resources for learning strategies, and choose the most appropriate cognitive strategies to solve problems according to specific situations.

The results of this study also show the differences in the development of students' cognitive strategies in mathematics. For example, in the "function" activity of this study, the teacher taught more advanced methods, but some students who had not been exposed to programming before did not master this method, so they still used their own methods in the subsequent task level, and in the cognitive development of arithmetic, all students can always complete the corresponding tasks according to the methods taught by the teacher. It shows that some students don't have the pre-concept of programming activities, so they can't finish coding and debugging according to the statement logic in programming in a short time. However, arithmetic activities are activities that students have been exposed to since childhood, and students can combine the relevant experience and pre-concept in the process of growth with the methods taught by teachers, and finally complete the corresponding tasks.

Therefore, on the whole, the development of children's cognitive strategies is complex and specific. Complexity is reflected in the fact that students do not completely abandon the low-level strategies formed by previous conceptual experience in the process of solving problems after learning more advanced strategies, but form a competitive relationship between low-level strategies and high-level strategies in the process of thinking. Although using more advanced strategies may increase the efficiency of solving problems, they are still in the process of accepting and understanding the more complicated and difficult advanced strategies they have just learned, which requires a lot of cognitive resources. It is impossible to complete a flexible application, so the specificity of cognitive strategy development is formed: students choose the most suitable strategy according to different environments to achieve the goal of solving problems.

There are also some shortcomings and defects in the design and implementation of this study. From the experimental link and process, this experiment focuses on the development process of students' algorithm strategy in programming activities, that is, based on whether they successfully complete the checkpoint task, while ignoring whether students have written wrong and redundant codes that will not affect the final task. Therefore, in the future research and development of this kind of programming software, the process can be designed more strictly, and if irrelevant code appears, it should also be prompted or challenged to fail, so as to ensure that students can successfully write and run the code not because of luck or subjective judgment. In addition, the selection of subjects in this experiment is not rigorous. Some students have already used Swift Playgrounds before taking part in the activity, so the results obtained in the pre-test stage and practice stage are not much different from those obtained in the post-test stage and migration stage, which leads to the unconvincing effect of the experiment.

References

Tedre, M., & Denning, P. J. (2016). The long quest for computational thinking. *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, 120–129.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Jeannette, M. W. (2006). Computational thinking. *COMMUNICATIONS OF THE ACM,* 49(3)

Winslow, B., Danielle, B. H., Katherine, J. N., Ken, P., Natalie, F., Camilla, N. J., Byron, L., Patrick, L., & Kasia, M. (2018). Active Learning Environments with Robotic Tangibles: Children's Physical and Virtual Spatial Programming Experiences. *TLT*, 11(1)

Saez-Lopez, J., Roman-Gonzalez, M., & Vazquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools [Article]. *COMPUTERS & EDUCATION*, 97, 129-141. http://doi.org/10.1016/j.compedu.2016.03.003

Zhao, W., & Shute, V. J. (2019). Can playing a video game foster computational thinking skills? *COMPUTERS & EDUCATION*, 141, 103633. http://doi.org/10.1016/j.compedu.2019.103633

Chen-Chung, L., Yuan-Bang, C., & Chia-Wen, H. (2011). The effect of simulation games on the learning of computational problem solving. Pergamon, 57(3)

Pellas, N., & Peroutseas, E. (2016). Gaming in Second Life via Scratch4SL: Engaging High School Students in Programming Courses. *JOURNAL OF EDUCATIONAL COMPUTING RESEARCH*, 54(1), 108-143. http://doi.org/10.1177/0735633115612785