Ouyang *et al.*
International Journal of STEM Education (2022) 9:45
https://doi.org/10.1186/s40594-022-00361-z

International Journal of STEM Education

RESEARCH Open Access

Applying multimodal learning analytics to examine the immediate and delayed effects of instructor scaffoldings on small groups' collaborative programming

Fan Ouyang*, Xinyu Dai and Si Chen

Abstract

Background: Instructor scaffolding is proved to be an effective means to improve collaborative learning quality, but empirical research indicates discrepancies about the effect of instructor scaffoldings on collaborative programming. Few studies have used multimodal learning analytics (MMLA) to comprehensively analyze the collaborative programming processes from a process-oriented perspective. This research conducts a MMLA research to examine the immediate and delayed effects of instructor scaffoldings on small groups' collaborative programming in K-12 education context with an aim to provide research, analytics, and pedagogical implications.

Results: The results indicated that the instructor provided five types of scaffoldings from the social, cognitive, and metacognitive dimensions, and groups had seven types of responses (i.e., immediate uptake and delayed use) to five instructor scaffoldings, ranging from the low-to-medium and high level of cognitive engagement. After the scaffolding was faded, groups used the content from the high-control cognitive scaffolding frequently to solve problems in a delayed way, but groups did not use the instructor's scaffolding content from the social and low-control cognitive scaffoldings from the pedagogical perspective, instructors should consider scaffolding types, group states and characteristics, as well as the timing of scaffolding to better design and facilitate collaborative programming. From an analytical perspective, MMLA was proved to be conducive to understand collaborative learning from social, cognitive, behavioral, and micro-level dimensions, such that instructors can better understand and reflect on the process of collaborative learning, and use scaffoldings more skillfully to support collaborative learning.

Conclusions: Collaborative programming is encouraged to be integrated in STEM education to transform education from the instructor-directed lecturing to the learner-centered learning. Using MMLA methods, this research provided a deep understanding of the immediate and delayed effects of instructor scaffoldings on small groups' collaborative programming in K-12 STEM education from a process-oriented perspective. The results showed that various instructor scaffoldings have been used to promote groups' social and cognitive engagement. Instructor scaffoldings have delayed effects on promoting collaborative programming qualities. It is highly suggested that instructors should integrate scaffoldings to facilitate computer programming education and relevant research should apply MMLA to reveal details of the process of collaboration.

*Correspondence: fanouyang@zju.edu.cn

College of Education, Zhejiang University, #866, Yuhangtang Rd., Hangzhou 310058, Zhejiang, China

Keywords: STEM education, Collaborative programming, Multimodal learning analytics, Collaborative analytics, Primary education

Introduction

In computer-supported collaborative learning (CSCL), small groups of students engage in higher level cognitive activities (e.g., ill-structured problem-solving) to create knowledge and relevant artifacts through sustained interactions, communications, and actions (Dillenbourg, 1999; Goodyear et al., 2014; Roschelle & Teasley, 1995). Such attributes manifest during collaborative computer programming where learners interact and collaborate with peers to learn programming languages, solve programming problems, and improve computational thinking (Demir & Seferoglu, 2020a; Plonka et al., 2015; Umapathy & Ritzhaupt, 2017). However, it is difficult for novice programmers to achieve a high quality of collaborative programming without external assistance in K-12 education (Hwang et al., 2012). Empirical studies have indicated that instructors usually need to provide scaffoldings, use scripted roles, and create a socially supportive atmosphere to improve the quality of collaboration (Fanchamps et al., 2021; Sun & Hsu, 2019).

However, empirical research shows discrepancies about the effect of instructor scaffoldings on collaborative learning. Prior empirical studies have indicated that major types of scaffoldings instructors include social scaffolding (e.g., Grossen & Bachmann, 2000; Perret-Clermont, 1980), motivational scaffolding (e.g., Gresalfi et al., 2009; Roehler & Cantlon, 1997) and cognitive scaffolding (e.g., Nedić et al., 2015; Tartas & Perret-Clermont, 2008). However, the instructor scaffoldings lead to different effects regarding the immediate uptake (the group completes the ongoing task) and delayed use (the group solves problems after the scaffolding is faded) in CSCL (Barron, 2003; Wittwer & Renkl, 2008). After reviewing literature, we find a lack of mature standard to evaluate the immediate and delayed effect of instructor scaffoldings on collaborative programming (Grévisse et al., 2019; Zheng et al., 2022). Moreover, most previous research have collected and analyzed a single data source (e.g., discourses) to understand the process of collaboration (e.g., van de Pol et al., 2019). Compared to the prior approach, the emergence of multimodal learning analytics (MMLA) has potential to better take advantage of multi-source data and comprehensively analyze the collaborative programming processes (Damşa & Nerland, 2016; Socratous & Ioannou, 2021; Sun et al., 2021). Moreover, the analytics of immediate and delayed effects is beneficial to reveal details of how instructor scaffoldings influence groups' collaborative programming from a process-oriented perspective (Brown & Renshaw, 2009). Therefore, it is necessary to further conduct a MMLA research to examine how the instructor scaffoldings influence the group's immediate uptake and delayed use from a temporal perspective, with an aim to provide research, analytics, and pedagogical implications for collaborative programming in K-12 education.

This research aims to empirically examine the immediate and delayed effects of instructor scaffoldings on small groups' collaborative programming in K-12 education. To achieve this goal, this research collects multimodal data from collaborative programming and uses MMLA (including click stream analysis, class video analysis, and quantitative content analysis) to analyze small groups' collaborative programming in China's elementary education. Based on empirical research results, this research proposes pedagogical and analytical implications for the practice and research of instructor scaffolding on collaborative programming education.

Literature review

Computer programming has been widely studied in computer science education, which can be considered as a sub-branch of science, technology, engineering and mathematics (STEM) education (Fedorenko et al., 2019; Guzdial & Morrison, 2016). Collaborative programming, as a CSCL mode, supports two or more learners working together at one workstation or remotely online to solve the same programming problems (Zheng, 2021). Collaborative programming enables students to interact and communicate with their peers while learning programing languages and debugging codes (Serrano-Cámara et al., 2014). Empirical studies have indicated that collaborative programming is beneficial to improve students' computer programming skills (Lin et al., 2018), to establish collective understanding of problems (Teague & Roe, 2008), and to develop computational thinking skills (Wei et al., 2020). However, collaborative programming requires students to use collaborative problem-solving strategies, higher-order programming logics and computational thinking skills, which usually bring up additional obstacles and difficulties, especially for novice students without sufficient programming knowledge and skills (Wang & Hwang, 2017). Major obstacles include establishing a positive collaborative relationship (Demir & Seferoglu, 2020b), sustaining a high-level knowledge construction (Papadakis, 2018), and making continuous knowledge innovation to solve problems (Veerasamy et al., 2019).

Therefore, groups usually need external assistance or support from the instructor to achieve a high quality of collaborative programming and adjust the subsequent learning (e.g., Tohyama et al, 2018; Zheng, 2021).

From the theoretical and practical perspective, instructor scaffoldings are used to facilitate collaborative learning process and performance (Hmelo-Silver & DeSimone, 2013; Kirschner et al., 2006; Stahl, 2009), particularly for STEM education. Primary scaffoldings include cognitive, metacognitive, or social scaffoldings. Cognitive scaffolding aims to facilitate the groups' knowledge-construction or problem-solving when the instructor provides additional explanations of the learning tasks, content or problems to be solved (Sandoval & Reiser, 2004; Zheng, 2021). Cognitive scaffolding can range from a low- to a high-level control, such as asking an open-ended question to directly providing answers. Second, metacognitive scaffolding aims to facilitate learners' planning, monitoring, and reflection of peers' and their own learning (Johnson, 2019; Kiemer et al., 2015; Kim et al., 2022). Research results show that the instructor's metacognitive scaffolding is beneficial to help groups reflect on the collaborative learning, enhance group awareness, and lead groups to transition from instructor regulation to group regulation (Kwon et al., 2013; Ouyang et al., 2022; Rasku-Puttonen et al., 2003). Third, social scaffolding aims to facilitate social interactions and promote social emotions during collaborative learning (Belland et al., 2017; Ouyang & Xu, 2022). Instructors use temperate social scaffolding to resolve group interpersonal disputes, to create a positive collaborative environment, and to encourage students to share their thoughts. Collaborative programming requires students utilize their cognitive, affective and social competencies to facilitate the group's social interactions, decision-making and problem-solving process in order to construct programming and assess the outcomes (Clark & Sengupta, 2020; Lai & Wong, 2022; Lavonen et al., 2002). Therefore, instructor scaffolding is a necessity during collaborative programming, that have been showed positive influences on groups' collaborative learning (Socratous & Ioannou, 2022).

The instructors' use of different types of scaffoldings has different effects on collaborative learning, usually reflected by the group's immediate uptake and delayed use of scaffoldings during collaboration (van de Pol et al., 2019; Webb & Farivar, 1999; Webb et al., 2006). The immediate effect of the instructor scaffolding aims to help groups reduce cognitive load, ease interpersonal conflicts, and adjust the collaborative process, in order to complete the ongoing tasks (van de Pol et al., 2010). The delayed effect of the scaffolding is reflected by the group's integration and application of the information into existing knowledge schemes to solve problems after the instructor's scaffolding is faded (Webb & Farivar, 1999; Wittwer & Renkl, 2008). Empirical studies have indicated a positive relationship between groups' immediate uptake to scaffoldings and the groups' collaborative experiences or performances. For example, Grévisse et al. (2019) used a scaffolding platform to help students integrate relevant materials in a programming environment and examined the effect of the scaffolding. The results showed that compared to the traditional learning process, students actively used scaffolding to increase participation in collaborative programming and highly appraised the scaffolding. Lin et al. (2018) employed a cooperative programming editing tool as a scaffolding to teach programming. The results showed that compared with the control group without scaffolding support, the groups that received the scaffolding were more prone to participate in and reflect on collaborative programming. Moreover, related studies have focused on investigating how students integrate and apply the contents of the instructor scaffoldings in a delayed way and empirically examine its delayed effect on collaboration. For example, van de Pol et al. (2019) found that groups tended to formulate more accurate answers when they applied the instructor scaffoldings in subsequent group tasks, compared to the groups that ignored the instructor's scaffolding. However, few studies have been located to explore how groups' delayed responses to the instructor's scaffolding influence their subsequent learning in collaborative programming. Therefore, it is necessary to further examine the immediate and delayed effects of instructor scaffoldings on collaborative programming and provide instructional strategies to improve collaborative programming quality based on empirical research results.

From an analytical perspective, with the development of MMLA, researchers can take advantage of audio, video, and other forms of multimodal data to understand the discourses, emotions, and behaviors of students in the group, which is conducive to gain a comprehensive understanding of collaborative programming. For example, Damşa and Nerland (2016) collected video-recordings of discussions, online messages, semi-structured interviews and final programming products, and analyzed students' participation, characteristics of the inquiry tasks, and use of knowledge resources. Sun et. al. (2021) used the click stream analysis, classroom video analysis, and discourses analysis to identify pairs' programming behaviors, discourses, and perceptions. Socratous and Ioannou (2021) collected and analyzed classroom video recordings, tablet screen-recordings and audio recordings and post-debugging tests data on collaborative programming, to compare the effect of a structured and an unstructured educational robotics curriculum. Compared to analytics driven by a single data

source, MMLA can better enrich the data collections of students' learning processes to help researchers understand the processes and environments of collaborative learning (Chatti et al., 2017; Di Mitri et al., 2021; Samuelsen et al., 2019). Moreover, multiple learning analytical methods are usually used in MMLA to analyze those data from multiple sources, including click stream analysis (e.g., Sun et al., 2021), video analysis (e.g., Khan, 2017), content analysis (e.g., Socratous & Ioannou, 2021), and discourse analysis (e.g., Gaul & Kim, 2020). In addition, collaborative programming is a dynamic process, consisting of a gradual changing process of problem-solving, meaning-making, and knowledge-construction (Lewis, 2012; Sun et al., 2021; Wu et al., 2019). Therefore, it is necessary to examine student groups' discourses, in-class behaviors, and computer operations from a process-oriented perspective, to fully understand the details of collaborative programming. Previous studies have examined the temporal change of the groups' verbal and embodied actions in STEM education (Riikonen et al., 2020), the student–student and student–instructor interactions during collaborative process (Kynigos & Diamantidis, 2022) and the groups' gaze patterns during pairs programming processes (Villamor & Rodrigo, 2019). More importantly, the focus of the current research is to examine the immediate uptake and delayed use of instructor scaffoldings, which in particular focuses on the temporal attributes and requires a detailed examination of the change process (Gidalevich & Kramarski, 2019; Holmes et al., 2014; Jadallah et al., 2011). Therefore, the temporal analysis is beneficial to comprehensively analyze and understand the influence of instructor scaffoldings on collaborative learning (Bloome et al., 2009; Brown & Renshaw, 2009; Wiig et al., 2017). In summary, it is necessary to collect multimodal data and use multimodal learning analytics from a temporal perspective to examine the effects of instructor scaffoldings on collaborative programming.

The current study

To address those research and practice gaps, this research used MMLA to investigate the immediate and delayed effects of instructor scaffoldings on small groups' collaborative programming. The overarching research purposes were twofold: (1) to improve the quality of collaborative programming through using instructor scaffoldings, and (2) to empirically examine the immediate and delayed effects of instructor scaffoldings on collaborative programming.

To achieve these purposes, this research used MMLA to collect multimodal data from students' collaborative process and analyze small groups' collaborative programming process before, during and after instructor scaffoldings. The research questions were: (1) *How did the instructor use five types of scaffoldings during the groups' collaborative programming process?* (2) *What were the immediate effects of instructor scaffoldings on collaborative programming?* and (3) *What were the delayed effects of instructor scaffoldings on collaborative programming?* Specifically, multimodal process data were collected, including audio data, class video data and computer screen-recording data. Multiple learning analytics methods were used, including primary analysis (click stream analysis, video analysis, content analysis, and qualitative micro-level analysis) as well as secondary analysis (statistical analysis, social network analysis, epistemic network analysis and temporal analysis), to demonstrate the immediate and delayed effects of different instructor scaffoldings. According to these results, this research proposed pedagogical and analytical implications for future practices of instructor scaffoldings in collaborative programming.

Methodology

Research context and participants

The research context was an extracurricular course titled "*Artificial Intelligence in Maker Education*" offered at an elementary school during Spring 2021 in the Eastern area of China. The subject of this course was one of the curricula in K-12 primary education in China (Chinese Ministry of Education, 2019). AI education helps K-12 students understand what the emerging AI technologies are and how they work. This course designed the computer programming practice to deepen students' understandings of AI-related knowledge and applications (Chiu, 2021; Pedró et al., 2019). Kittenblock, a graphical programming environment based on Scratch, was used in this course to enhance novice programmers' complex problem-solving and computational thinking skills (Tohyama et al., 2018). The course was taught by a female instructor, who was a doctoral student in the educational technology program at a top research-intensive university in China. She had 1 year of teaching experience in K-12 education and good understandings of different types of scaffoldings in collaborative learning. Twenty-eight 3rd to 5th graders (9 females, 19 males) enrolled in this course. All students were new to the course content, and new to computer programming. The instructor randomly arranged students into nine groups (three students/group) at the beginning of the course and the groups were fixed throughout the course.

The instructor designed two instruction and learning phases throughout 11 weeks (see Fig. 1), namely the instruction and learning phase (including instructor lectures and collaborative programming activities) (Week 1–8), and the project-based learning (PBL) phase

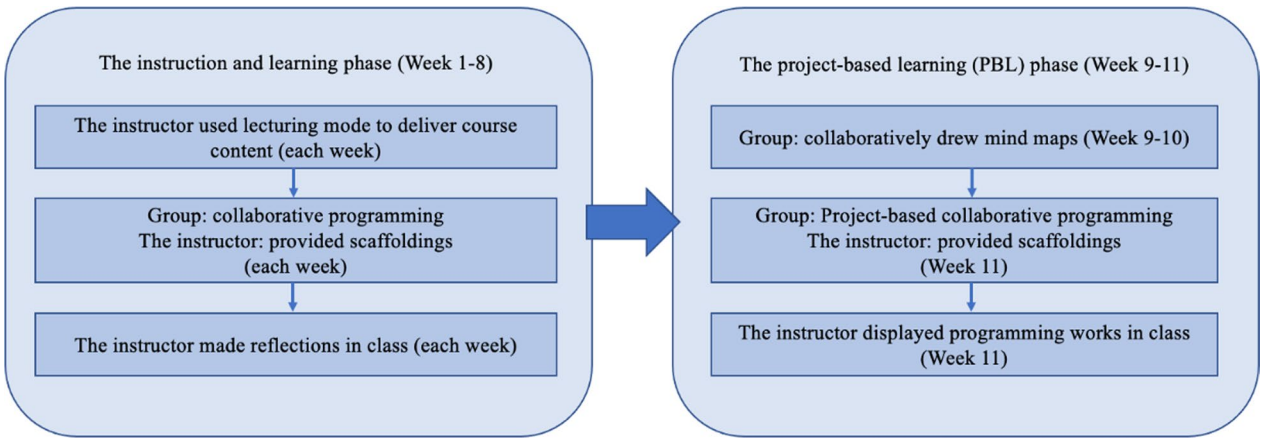

Fig. 1 The instructional procedure

(including collaborative mind mapping and project-based collaborative programming activities) (Week 9–11). During Week 1–8, all groups were required to complete eight collaborative programming tasks, such as text recognition, speech recognition, and image recognition, etc. During Week 9–11, all groups applied what they learned in Week 1–8 to design and produce their own innovative programming works, such as anti-fraud smartphones, smart glasses and smart crutches.

In the instruction and learning phase (Week 1–8), the instructor used the traditional lecturing mode to deliver course content (see Fig. 2a). Then the instructor explained the collaborative tasks to students and asked the groups to discuss group plans to complete tasks (see Fig. 2b). Each group conducted the collaborative programming on one shared computer and used Kittenblock software, a block-based visual programming software to complete tasks (see Fig. 2d). During Week 1–8, each group completed eight tasks in total, including voice recognition, license plate recognition, color recognition, etc. In the PBL phase (Week 9–11), each group applied the knowledge learned in the previous lectures and used programming skills to design and make a product (such as anti-fraud smartphone, smart magnifying glass and smart crutches, etc.). In Week 9–10, the groups designed the product through drawing a mind map collectively; the purpose was to discuss and determine the user requirements, product functions, programming modules and required materials. In Week 11, the groups carried out collaborative programming based on the mind map and completed the programming task. Throughout the course (Week 1–11), when the groups discussed or conducted collaborative programming, the instructor walked around the classroom to observe and provided scaffoldings in terms of the groups' current states (see Fig. 2c).

Data collection and analysis approaches

This research collected and analyzed multimodal data through three ways (see Fig. 3). First, four cameras were placed in four different positions in the classroom to record the in-class behaviors of the instructor and all students. Then, the computer screens of nine groups were recorded to acquire students' clickstreams of computer programming operations. To ensure the consistency of multimodal data sources, the current research used in-class video data, discourse audios, and computer screen-recording data during collaborative programming (Week 1–8 and Week 11). About 18 h of in-class video data (without audio) and 32 h of computer screening data (with audio) were collected and analyzed. Students were informed about the research purpose and data collection process; all of them agreed the data collection and signed consent forms before the course started with the assistance from the school teachers.

Referring to the MMLA in CSCL as the rationale (Cress et al., 2021), primary analysis methods included click stream analysis, video analysis, and content analysis. There were three steps in the analysis process. The first step was to transcribe class video, computer screening and audio data into excel files in the chronological order. There were initially 81 files (nine groups * 9 weeks), of which 62 contained the instructor scaffoldings. Each file recorded the instructor and a group's conversational discourses, computer programming operations, and in-class behaviors in each class. Moreover, 5-s time interval was used as the unit for transcription. The time-based segmentation gave a structured unit for analysis and also allowed a temporally unfolding overview of the group collaborative learning (Sinha et al., 2015). In most cases of our data, the 5-s interval was enough for the students to speak a complete sentence, produce a physical behavior,

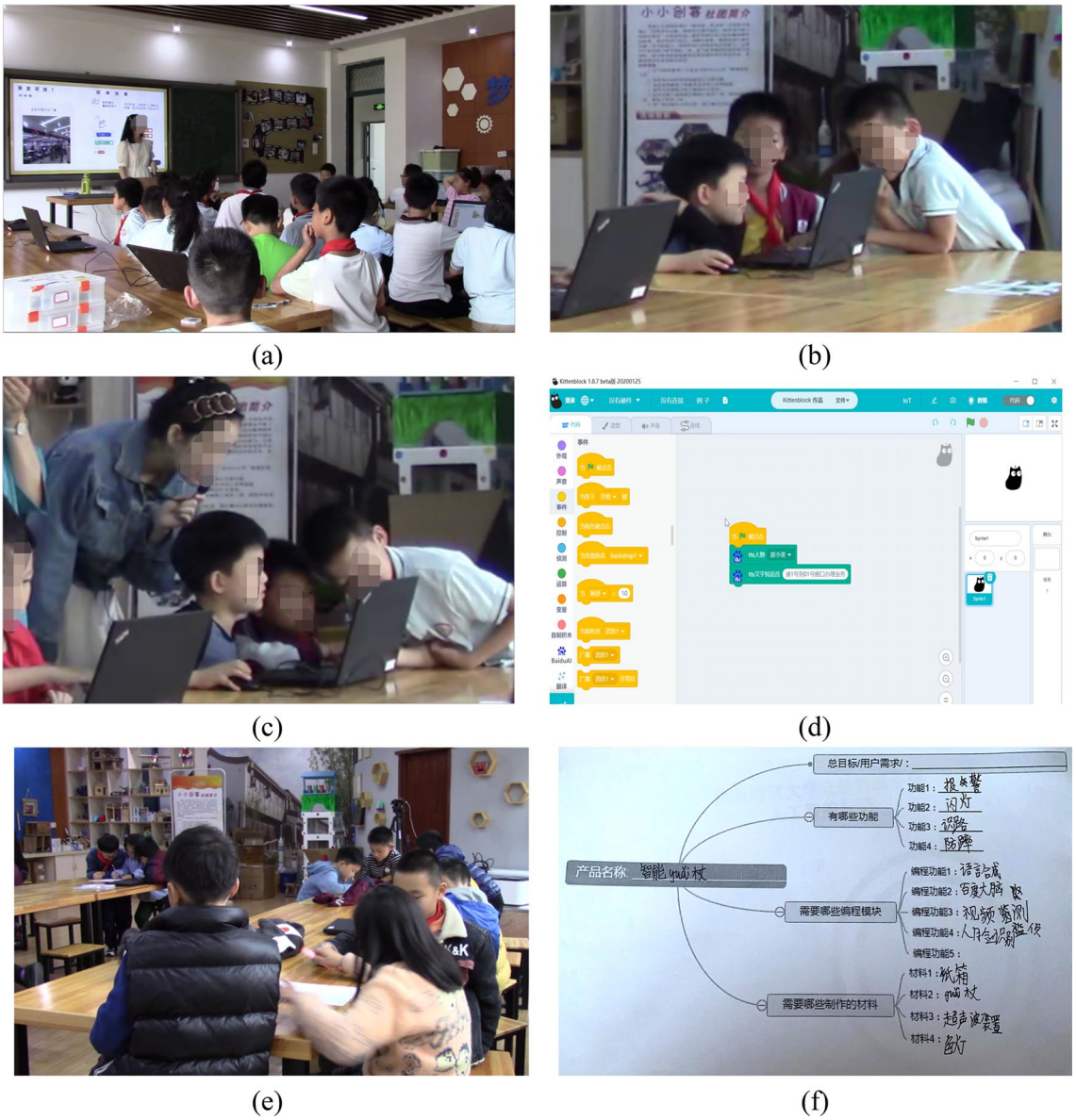

Fig. 2 In-class photos and computer screenshots. **a** The instructor's lectures; **b** collaborative programming; **c** the instructor provided scaffoldings; **d** a screenshot of the Kittenblock software; **e** collaborative drawing of mind maps; and **f** a completed mind map by a group

or complete a computer programming operation. Next, we divided all multimodal data into three phases with a 20-s segment (see Fig. 3): (1) Phase 1: 20 s before the instructor provided a scaffolding; (2) Phase 2: 20 s after the instructor provided a scaffolding, as the range of the immediate uptake; and (3) Phase 3: 20 s after Phase 2 until the end of a class, as the range of the delayed use of instructor scaffolding. The rationale for choosing 20-s as the interval depended on cases generated from our data.

In the second step, content analysis, video analysis, click stream analysis methods were used to code the transcribed data. Two coders first individually watched the computer screening and the in-class videos to proofread the excel file, and then determined the initial three coding frameworks, namely collaborative group states, instructor scaffoldings and groups' immediate uptakes. Then two coders had discussions to achieve an agreement of the final coding frameworks of the collaborative group states (see Table 1), instructor scaffoldings (see Table 2) and groups' immediate uptakes of instructor scaffoldings (see Table 3). There were two considerations in the coding process. First, for the instructor scaffoldings, if an

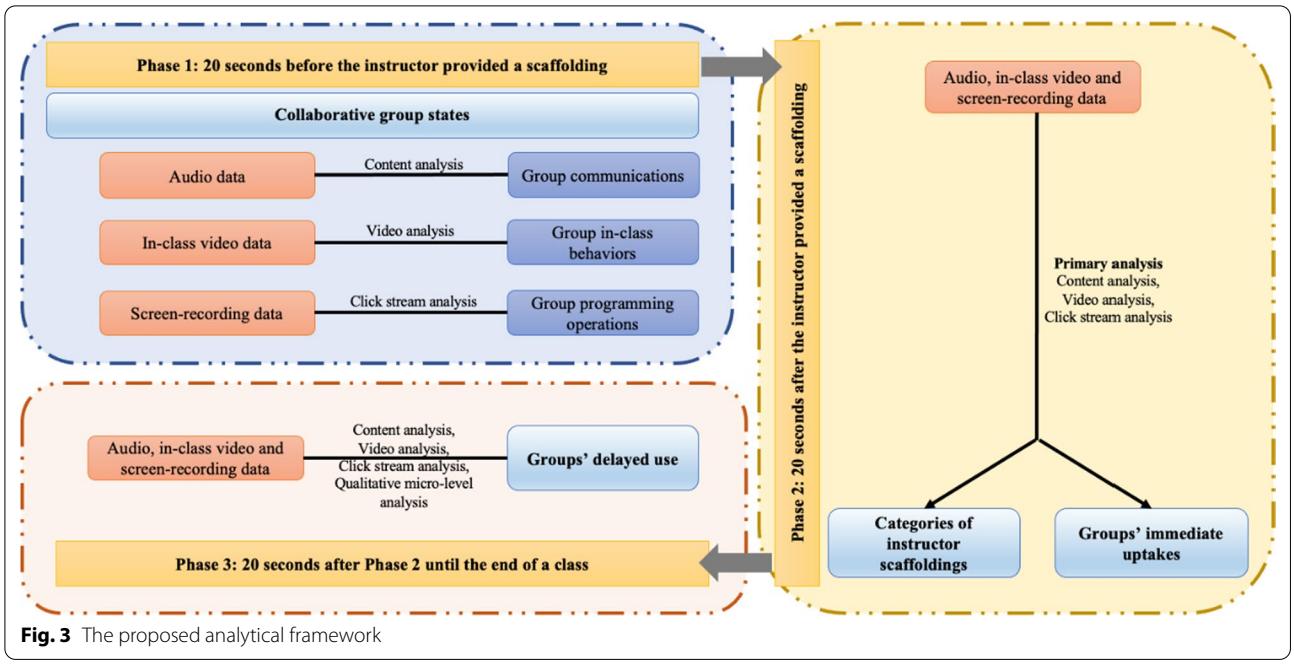

Fig. 3 The proposed analytical framework

Table 1 The coding framework for analyzing collaborative group states

Code	Description
Programming operation (PO)	The group member collaboratively programmed on the shared computer
Asking assistance from the instructor (AA)	The group member asked assistance from the instructor either verbally or physically
Group discussion (GD)	The group member had discussions with each other about collaborative tasks
Group conflict (GC)	The group member had conflicts that were difficult to reconcile

Table 2 The coding framework for instructor scaffoldings (c.f. van de Pol et al., 2019)

Category	Code	Description
Social dimension	Social scaffolding (SS)	The instructor helped the groups to establish the collaborative atmosphere with a sense of trust, support, and security
Cognitive dimension	Low-control cognitive scaffolding (CS-L)	The instructor raised open-ended questions that elicited the groups' thinking without providing new information
	Medium-control cognitive scaffolding (CS-M)	The instructor provided hints or clues to help the groups solve cognitive problems
	High-control cognitive scaffolding (CS-H)	The instructor provided answers directly to the group or used the computer to show how to program
Metacognitive dimension	Metacognitive scaffolding (MS)	The instructor planned, monitored and regulated the group's learning goals and collaborative processes

instructor scaffolding was longer than 5 s, only the last instructor scaffolding which the group reacted to was coded. Second, immediate uptakes from students in the same group were all coded. For example, the same immediate uptakes from two students within the group were recorded twice. Then, two coders independently coded the data again in a chronological order based on the final coding frameworks, and reached an inter-rater reliability with the Cohen's Kappa of 0.872.

Finally, based on the primary analysis of coding, the secondary analysis was used to demonstrate the immediate and delayed effects of instructor scaffoldings on collaborative programming, including statistical analysis (SA), social network analysis (SNA),

Table 3 The coding framework of groups' immediate uptakes of instructor scaffoldings (c.f. Barron, 2003)

Code	Description
Keeping silent to scaffolding (KPS)	The group kept silent or maintained no actions to the instructor scaffolding, e.g., did not communicate nor conducted programming on the computer
Ignoring scaffolding (IGN)	The group did not mention or use the instructor scaffolding and continued to try to solve the problem according to their own thoughts
Refusing scaffolding (REF)	The group explicitly refused to refer to the instructor scaffolding
Repeating or copying scaffolding (REC)	The group repeated or copied the content of the instructor scaffolding
Asking questions about scaffolding (ASQ)	The group asked the instructor a question when they did not understand the content of the instructor scaffolding
Responding to scaffolding (RES)	The group simply responded to the instructor scaffolding, e.g., a student answered the instructor's question with "yes" or "no"
Understanding or applying scaffolding (UAS)	The group elaborated on the content of the scaffolding in their own words or applied it to solve problems

epistemic network analysis (ENA), and temporal analysis (TA) (see Fig. 4). Regarding the effect of immediate uptake, SA (including analysis of variance, analysis of covariance, and Sankey visualization) was used to demonstrate the overall transitional frequencies from the collaborative group states to instructor scaffoldings and to groups' immediate uptakes. SNA was used to examine the social networks of instructor–student interaction and student–student interaction after the instructor scaffoldings were provided based on groups' in-class behaviors, computer operations and communication discourses. ENA was used to examine the overall structure of group's immediate uptakes based on co-occurrence relationships of codes and was visualized in a network model to demonstrate the immediate effect (Shaffer & Ruis, 2017; Shaffer et al., 2009, 2016). Qualitative micro-level analysis was used to examine the delayed use of the groups to instructor scaffoldings based on groups' in-class behaviors, computer operations and communication discourses, and TA was used to visualize the micro-level process in the form of time series diagrams.

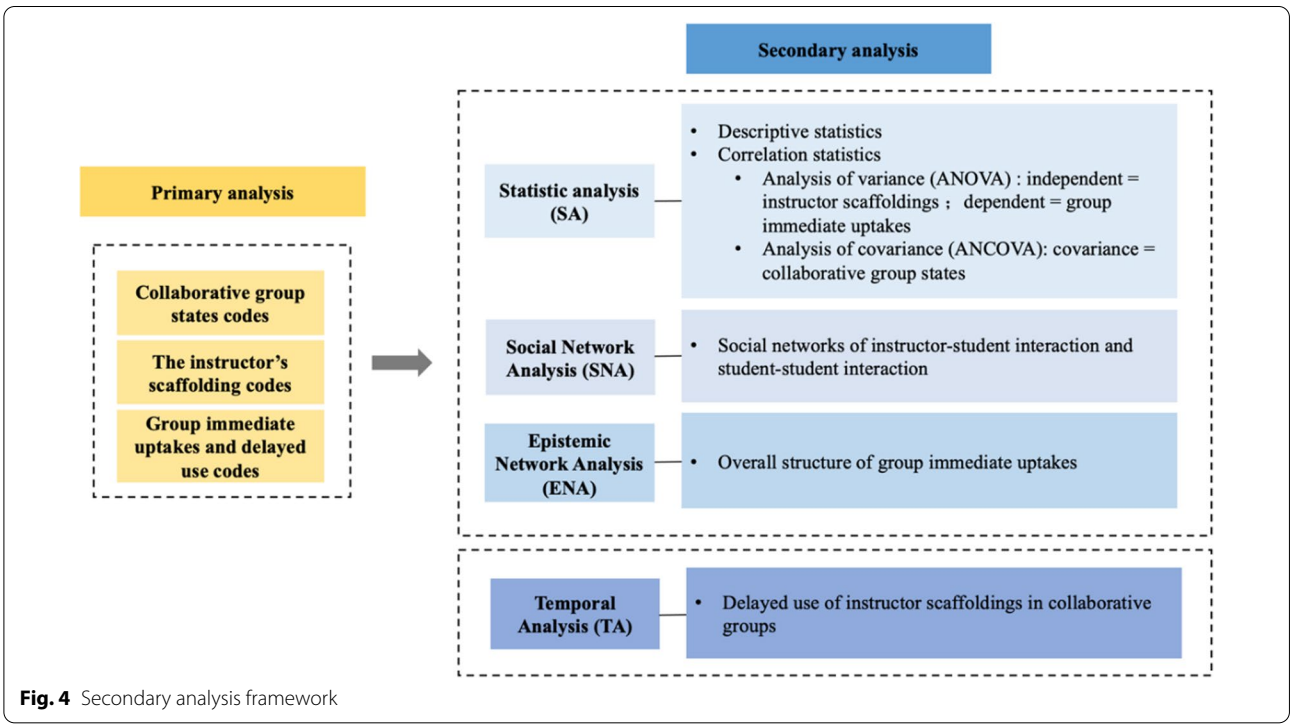

Fig. 4 Secondary analysis framework

Results

RQ1: How did the instructor use five types of scaffoldings during the groups' collaborative programming process?

The Sankey diagram demonstrated the transitional frequencies from collaborative group states, to instructor scaffoldings, and to groups' uptakes of scaffoldings (see Fig. 5). The instructor provided a total frequency of 919 scaffoldings. For the instructor scaffolding codes, there were 84 codes of social scaffolding (SS) (9.14%), 225 codes of metacognitive scaffolding (MS) (24.48%), 231 codes of low-control cognitive scaffolding (CS-L) (25.14%), 219 codes of medium-control cognitive scaffolding (CS-M) (23.83%), and 160 codes of high-control cognitive scaffolding (CS-H) (17.41%). For the group state codes, there were 144 codes of group discussion (GD) (40.11%), 141 codes of programming operation (PO) (39.28%), 62 codes of asking assistance from the instructor (AA) (17.27%), and 12 codes of group conflict (GC) (3.34%). When the group discussed collaborative tasks (GD), the instructor used metacognitive scaffolding (MS) (Freq. = 46) most frequently, followed by the medium-control (CS-M) (Freq. = 37) and low-control (CS-L) (Freq. = 36) cognitive scaffoldings. When the group was in a collaborative programming state (PO), the instructor used metacognitive scaffolding (MS) (Freq. = 56) most frequently, followed by low-control cognitive scaffolding (CS-L) (Freq. = 36). When the group asked assistance from the instructor (AA), the instructor used the low-control cognitive scaffolding (CS-L) (Freq. = 23) most frequently, followed by metacognitive scaffolding (MS) (Freq. = 14). When the groups had conflicts (GC), the instructor used social scaffolding (SS) (Freq. = 6), followed by metacognitive (MS) (Freq. = 4) and low-control cognitive (CS-L) (Freq. = 2) scaffoldings.

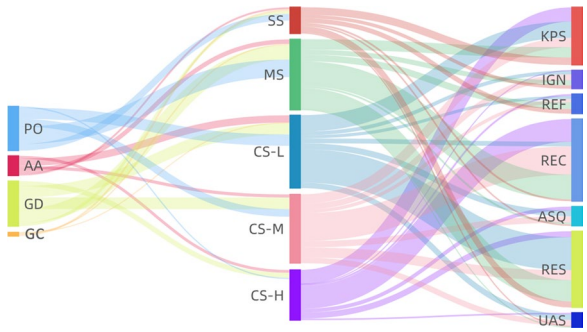

Fig. 5 The Sankey diagram of transitional frequencies

RQ2: What were the immediate effects of instructor scaffoldings on collaborative programming?

The immediate effects of instructor scaffoldings were examined from the perspectives of statistical significance, social interaction, and overall structure. First, from the statistical perspective, statistical analysis of the Shapiro–Wilk normality test, ANOVA and ANCOVA were conducted to examine the differences among five instructor scaffoldings (see Table 4). The normality of the distributions was first tested using the Shapiro–Wilk test and the result indicated that data were normally distributed ($p > 0.05$). Next, ANOVA results showed statistically significant differences among five instructor scaffoldings on all seven codes of immediate uptakes from groups, namely keeping silent to scaffolding (KPS) ($F = 5.12$, $p < 0.001$), ignoring scaffolding (IGN) ($F = 6.28$, $p < 0.001$), refusing scaffolding (REF) ($F = 4.43$, $p < 0.001$), repeating or copying scaffolding (REC) ($F = 28.47$, $p < 0.001$), asking questions about scaffolding (ASQ) ($F = 2.62$, $p < 0.05$), responding to scaffolding (RES) ($F = 19.87$, $p < 0.001$), and understanding or applying scaffolding (UAS) ($F = 6.99$, $p < 0.001$). The ANCOVA results showed that after taking *collaborative group state* as a covariate, there were statistically significant differences of five instructor scaffoldings on seven codes, namely KPS ($F = 5.15$, $p < 0.001$), IGN ($F = 6.40$, $p < 0.001$), REF ($F = 4.45$, $p < 0.001$), REC ($F = 28.44$, $p < 0.001$), ASQ ($F = 2.64$, $p < 0.05$), RES ($F = 19.98$, $p < 0.001$), and UAS ($F = 7.02$, $p < 0.001$).

After examining the average frequency of groups' immediate uptakes, it was found that, the groups had the highest frequency of RES (mean = 0.54, SD = 2.03) and UAS (mean = 0.15, SD = 0.88), two codes that reflected the high level of cognitive engagement, after the scaffoldings of CS-L and CS-M. Groups had the highest frequency of REC (mean = 0.69, SD = 1.86) and ASQ (CS-L: mean = 0.12, SD = 0.56; CS-M: mean = 0.12, SD = 0.87), two codes that reflected the medium level of cognitive engagement, after the scaffoldings of CS-L and CS-M. Groups had the highest frequency of KPS (mean = 0.40, SD = 0.68), IGN (mean = 0.27, SD = 0.38) and REF (mean = 0.20, SD = 2.04), Three codes that reflected the low level of cognitive engagement, after the SS scaffolding. In summary, groups showed a medium or high level of cognitive engagement after the scaffoldings of CS-L and CS-M, and a low level of cognitive engagement after the SS scaffolding.

Second, from the social interaction perspective, groups had different levels of the instructor–student interaction and student–student interaction after the instructor provided five scaffoldings (see Fig. 6). On average, groups had the highest level of overall instructor–student

Table 4 Descriptive statistics (mean, SD) of the group codes for five instructor scaffoldings

	SS ($N=84$)	MS ($N=225$)	CS-L ($N=231$)	CS-M ($N=219$)	CS-H ($N=160$)	ANOVA		ANCOVA		Pairwise comparisons
						F	*P*	*F*	*P*	
KPS	0.40 (0.68)	0.21 (0.79)	0.25 (1.23)	0.16 (0.96)	0.31 (1.76)	7.57	< 0.001***	7.619	< 0.001***	SS > CS-L > CS-M, SS > MS, CS-H > MS, CS-H > CS-M
IGN	0.27 (0.38)	0.10 (0.53)	0.07 (0.43)	0.14 (0.55)	0.03 (0.12)	9.37	< 0.001***	9.56	< 0.001***	SS > CS-M > CS-H, SS > MS > CS-H, SS > CS-L
REF	0.20 (2.04)	0.10 (0.66)	0.04 (0.65)	0.05 (0.46)	0.06 (0.29)	6.61	< 0.001***	6.641	< 0.001***	SS > MS > CS-L, SS > MS > CS-M
REC	0.11 (0.29)	0.57 (1.64)	0.14 (0.56)	0.69 (1.86)	0.59 (1.37)	42.43	< 0.001***	42.38	< 0.001***	MS > SS, MS > CS-L, CS-M > SS, CS-M > CS-L, CS-H > SS, CS-H > CS-L
ASQ	0.06 (0.58)	0.03 (1.39)	0.12 (0.58)	0.12 (0.87)	0.11 (0.61)	3.90	< 0.01**	3.927	< 0.01**	CS-L > MS, CS-M > MS, CS-H > MS
RES	0.27 (0.63)	0.35 (1.29)	0.54 (2.03)	0.16 (0.92)	0.14 (0.68)	29.64	< 0.001***	29.81	< 0.001***	MS > CS-L > SS > CS-H, MS > CS-L > SS > CS-M
UAS	0.06 (1.23)	0.03 (0.18)	0.13 (0.55)	0.15 (0.88)	0.05 (0.32)	10.46	< 0.001***	10.49	< 0.001***	CS-L > CS-H > MS, CS-M > CS-H > MS, CS-L > SS, CS-M > SS

*$p < 0.10$, **$p < 0.05$, ***$p < 0.001$

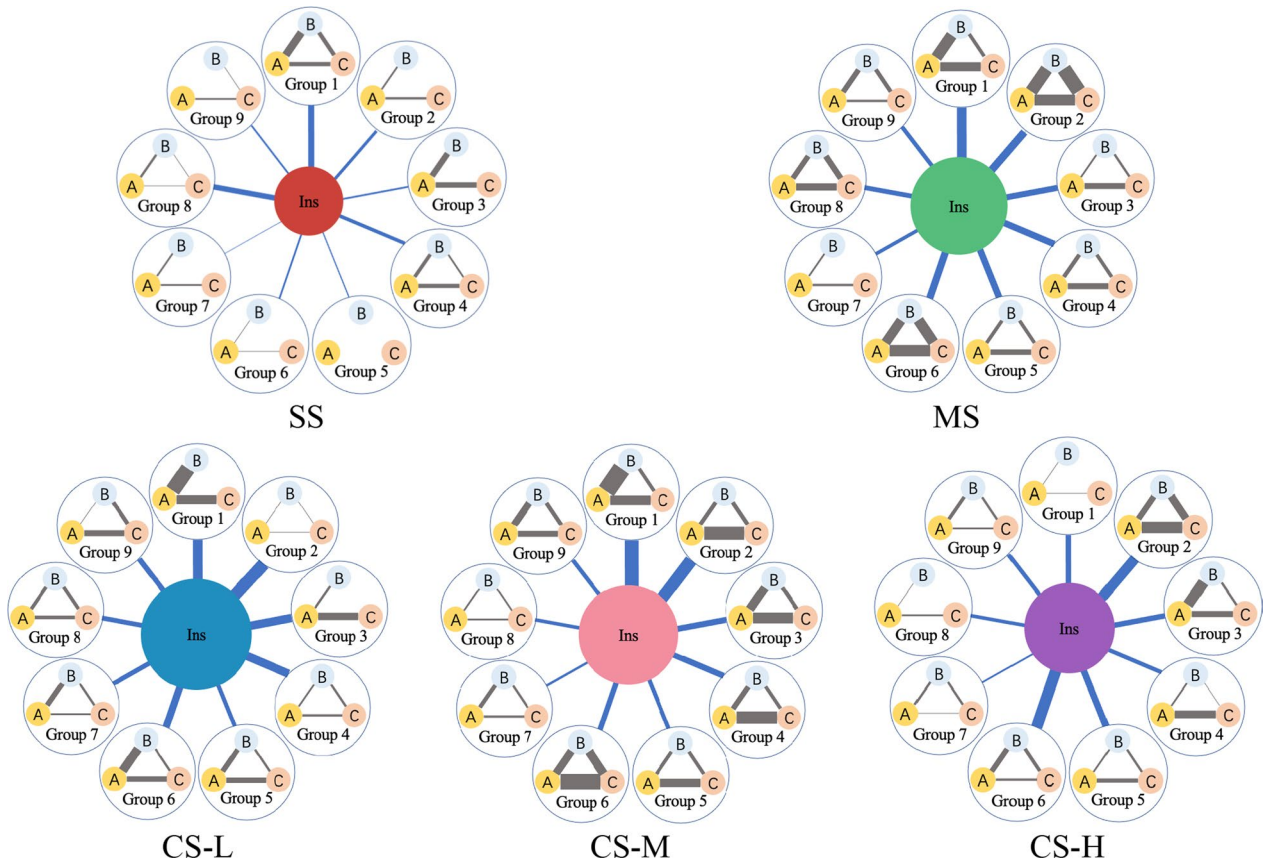

Fig. 6 The social networks of the five instructor scaffoldings. Node sizes of instructor represented the frequencies of instructor scaffolding; the link widths between two student nodes represented the interaction frequencies

interaction (mean = 49.44, SD = 17.33) after the instructor provided the low-control cognitive scaffolding (CS-L) (e.g., Group 2: Freq. = 82; Group 1: Freq. = 69; Group 4: Freq. = 57). Groups had the highest level of overall student–student interaction (mean = 23.33, SD = 10.22) after the instructor provided the medium-control cognitive scaffolding (CS-M) (e.g., Group 1: Freq. = 38; Group 6: Freq. = 37; Group 2: Freq. = 29). In addition, groups' overall instructor–student interaction ranked at the second place after the instructor provided the medium-control cognitive scaffolding (CS-M) (mean = 43.89, SD = 25.13). The groups had a medium level of overall instructor–student interaction (mean = 42.44, SD = 12.07) and overall student–student interaction (mean = 20.78, SD = 11.36) after the instructor provided metacognitive scaffolding (MS). And groups had a relatively low level of overall instructor–student interaction (mean = 39.11, SD = 18.07) and overall student–student interaction (mean = 11.89, SD = 8.23) after the instructor provided high-control cognitive scaffolding (CS-H). Finally, groups had the lowest level of overall instructor–student interaction (mean = 18.56, SD = 10.88) as well as the lowest level of overall student–student interaction (mean = 6.89, SD = 5.80) after the instructor provided social scaffolding (SS). In summary, groups had the highest level of the instructor–student interaction after CS-L, and the lowest level of instructor–student interaction after SS; moreover, groups had the highest level of student–student interaction after CS-M, and the lowest level of student–student interaction after SS.

Third, from the overall structure perspective, ENA plots of immediate uptake codes were used to demonstrate the structures after five instructor scaffoldings (see Fig. 7). Groups illustrated strong connections of KPS-REC (coefficient = 0.18) and REC-RES (coefficient = 0.16) after medium-control cognitive scaffolding (CS-M). Groups illustrated strong connections of KPS-REC (coefficient = 0.19) and REC-RES (coefficient = 0.19) after high-control cognitive scaffolding (CS-H). Therefore, similar connections occurred after the instructor's cognitive scaffoldings of CS-M and CS-H, namely the strong connections between silence (KPS) and repeating/copying (REC) codes and between repeating/copying (REC) and responding (RES) codes.

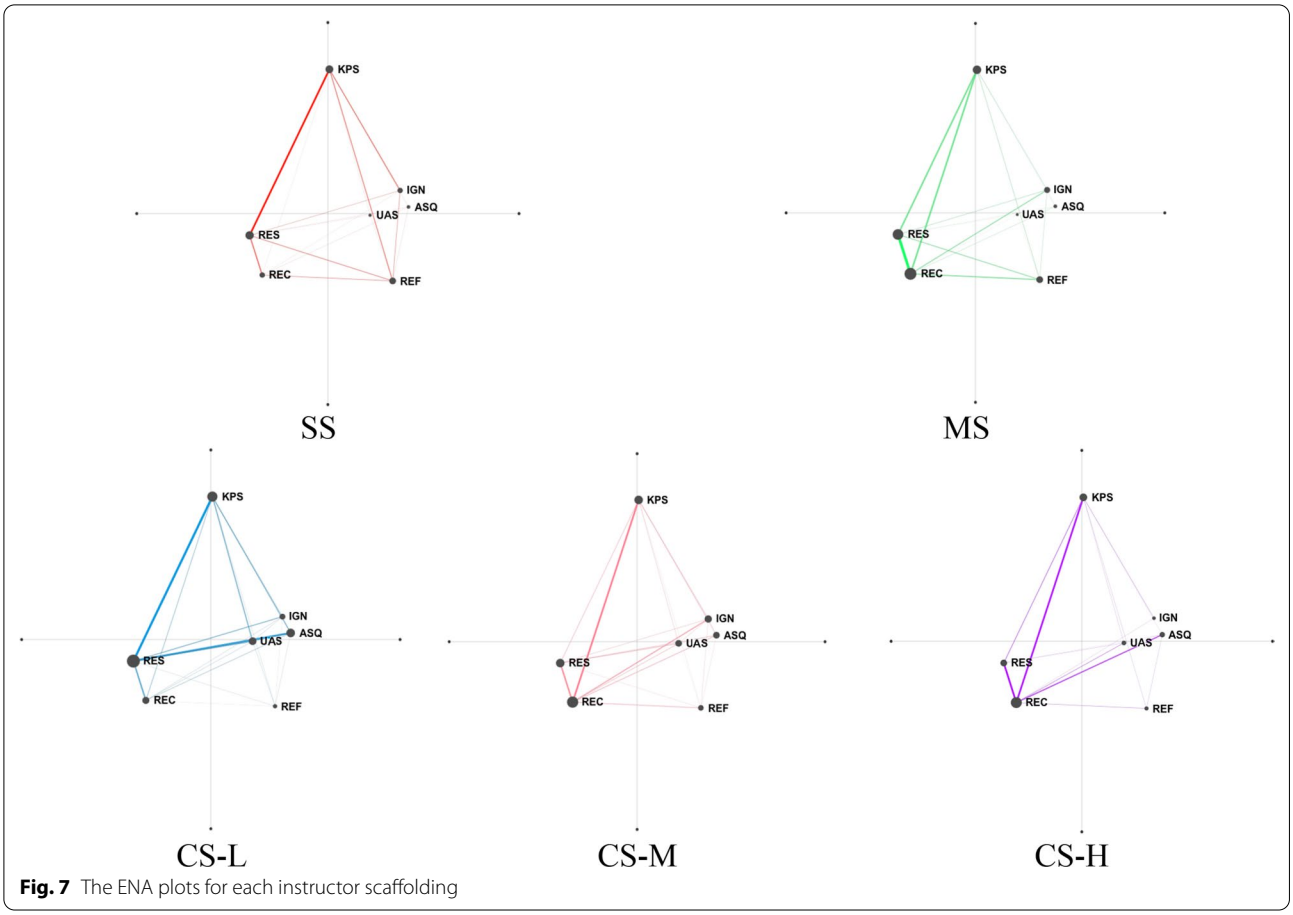

Fig. 7 The ENA plots for each instructor scaffolding

The results indicated that after the instructor provided the medium- and high-control cognitive scaffoldings, groups tended to keep silent and repeat the scaffolding content, and then simply respond to the scaffolding content. Compared with the CS-M and CS-H scaffoldings, groups had different connected structures after the CS-L scaffolding. Groups illustrated strong connections between the silence (KPS) and responding (RES) codes (coefficient = 0.23) and asking questions (ASQ) and responding (RES) codes (coefficient = 0.20) after low-control cognitive scaffolding (CS-L). That is, after the instructor provided the low-control cognitive scaffolding, the groups tended to respond to the scaffolding content after keeping silent or asking questions. In addition, groups illustrated a strong connection between the silence (KPS) and responding (RES) codes (coefficient = 0.18) after instructor's social scaffolding (SS). Finally, after the MS scaffolding, groups had strong connections between repeating/copying (REC) and responding (RES) codes (coefficient = 0.29) and between silence (KPS) and repeating/copying (REC) codes (coefficient = 0.18). In summary, groups' overall structures contained the highest frequent connections that involved higher level of cognitive engagement codes (e.g., ASQ, RES) after the MS and CS-L scaffoldings, but had the lowest frequent connections that involved higher level of cognitive engagement after the SS scaffolding.

RQ 3: What were the delayed effects of instructor scaffoldings on collaborative programming?

To understand the groups' delayed use of the instructor scaffoldings, we finally conducted a qualitative micro-level analysis supported with temporal visualizations based on groups' in-class behaviors, computer operations and communication discourses. The results showed that there were three types of instructor scaffoldings that were understood or used by groups in subsequent collaboration, including metacognitive scaffolding (MS) (Freq. = 2), medium-control cognitive scaffolding (CS-M) (Freq. = 4) and high-control cognitive scaffolding (CS-H) (Freq. = 5). Three exemplary fragments were selected here, namely Group 8 completed the color recognition task with the instructor's MS scaffolding, Group 7 completed the text recognition task with the instructor's CS-M scaffolding, and Group 6 completed the bank queue system with the instructor's

Table 5 The exemplified fragment of Group 8's delayed use of the MS scaffolding

Line	Time	Participant	Content	Code
1	19:00	Ins	If your group has no ideas, you can make some explorations to find inspiration	MS
2	19:05	S3	Okay. [Looking at computer screen]	RES
3		S1, S2	[Looking at computer screen and keeping silent]	KPS
4	19:10	S3	(Dragging the programming blocks of "repeat" into the interface and adding it to existing programs)	REC
5		S1, S2	[Looking at computer screen and keeping silent]	KPS
6	19:15	S3	(Modifying the parameter value of "value = = 0" to "value = = 20")	REC
7		S1, S2	[Looking at the computer screen and keeping silent]	KPS
8	19:20	S3	(Clicking the "Run" button, the computer successfully recognizes the color)	REC
9		S1, S2	[Looking at the computer screen and keeping silent]	KPS
The instructor made some reflections in class and assigned a new task, i.e., identifying other colors				
The group members discussed verbally; members did not conduct programming behaviors				
10	27:35	S2	Let's try to implement the function of recognizing blue. You guys can try programming first and let me think about it	UAS
11		S1, S3	[Looking at the computer screen and keeping silent]	KPS
12	27:40	S1, S2, S3	[Looking at the computer screen and keeping silent]	KPS
13	27:45	S2	Okay, let me try it	REC
14		S1, S3	Okay	RES
15	27:50	S2	(Setting the state of the initial color to transparent color, and after running the program, the computer successfully recognizes the new color)	UAS
16		S1, S3	Perfect	RES

Texts in "()" indicated that group students' computer programming operations; texts in "[]" indicated students' in-class behaviors; other texts were transcribed communication content

Table 6 The exemplified fragment of Group 7's delayed use of the CS-M scaffolding

Line	Time	Participant	Content	Code
1	27:20	Ins	To initialize the settings first, and then extract the image features	CS-M
2	27:25	S1, S2, S3	(Still programming according to their own ideas. Adding the "extract features" programming block to the existing program) [ignoring instructor' scaffolding]	IGN
3	27:30	S1, S2, S3	(Clicking the "Run" button, but the computer does not recognize the text) [ignoring instructor' scaffolding]	IGN
4	27:35	S1	(Looking for the parameter that controls the image for initialization)	REC
5		S2, S3	[Looking at the computer screen and keeping silent]	KPS
6	27:40	S1	(Clicking the "Run" button, and the computer successfully recognized the text)	REC
7		S2, S3	[Looking at the computer screen and keeping silent]	KPS
The instructor made reflections in class and assigned a new task, i.e., identifying a new idiom				
The group programmed and debugged many times but the computer still displayed the results from the previous task, and no new idioms were recognized				
8	33:00	S1	Because we do not re-initialize the settings, there is no way to recognize the new image	UAS
9		S2, S3	[Nodding to show agreement]	RES
10	33:05	S3	You are right, now we need to initialize	UAS
11		S1, S2	Okay	RES
12	33:10	S1	Let's set the background color to a transparent color, and initialize it (Finding the initial parameter and set it)	UAS
13		S2, S3	[Looking at the computer screen and keeping silent]	KPS
14	33:15	S2	Transparent color, choose this! [Pointing out the required parameters on the computer screen]	UAS
15		S1, S3	[Looking at the computer screen and keeping silent]	KPS

CS-H scaffolding (see Tables 5, 6, 7, Figs. 8, 9, 10). In Group 8, the instructor prompted the group to advance the collaborative task through exploratory activities, and in subsequent collaborative learning (about eight minutes after the instructor provided the scaffolding), the group actively engaged in exploratory activities on the task

Table 7 The exemplified fragment of Group 6's delayed use of the CS-H scaffolding

Line	Time	Participant	Content	Code
1	31:00	Ins	If you want to implement the calling function of the banking system, you have to find the parameter that controls the serial number and set it to 1. (Finding and modifying the parameter on the computer)	CS-H
2	31:05	S1, S2, S3	[Looking at the computer screen and keeping silent]	KPS
3	31:10	S1, S2, S3	[Looking at the computer screen and keeping silent]	KPS
4	31:15	S1, S2, S3	[Looking at the computer screen and keeping silent]	KPS
5	31:20	S1, S2, S3	[Looking at the computer screen and keeping silent]	KPS
The instructor required the group to implement the function of automatic increase of numbers				
After the group failed in debugging, they found that the control parameters were not set, and they repeated the previous steps as the instructor pointed out				
6	36:05	S2	We do not set this parameter to 1, and the reported number will not change when the program is running	UAS
7		S1, S3	[Nodding to show agreement]	RES
8	36:10	S2	(Finding and dragging the programming blocks)	REC
9		S1, S3	[Looking at the computer screen and keeping silent]	KPS
10	36:15	S3	Yes, we have to find the corresponding parameters first, and then set them according to our requirements	UAS
11		S1, S2	[Nodding to show agreement]	RES
12	36:20	S3	I see it, this is the parameter	REC
13		S1	Yes	PRS
14		S2	[Nodding to show agreement]	RES

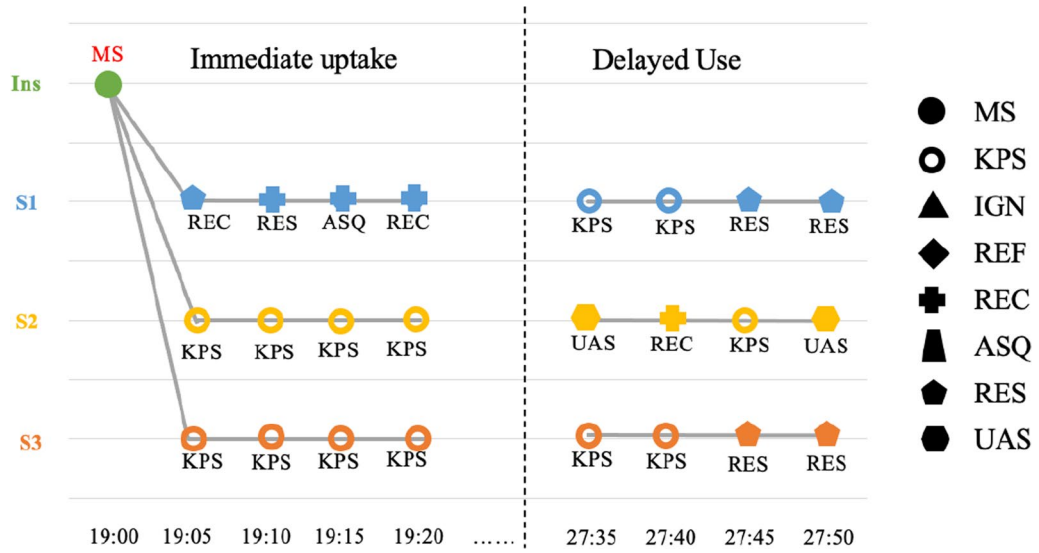

Fig. 8 The temporal transition after the MS scaffolding

without external assistance. Specifically, S3 was the most active student after the instructor provided the metacognitive scaffolding (MS). After a quick response, she accepted and followed instructor's advice and conducted an exploratory programming (see Table 5: line 2, 4, 6, 8). In the subsequent learning (i.e., the delayed use of scaffolding), S3 initially showed silence to the new collaborative task and the suggestion presented by peers (see Table 5: line 11, 12), and then S3 verbally responded to the peer's opinions and programming operations (see Table 5: line 14, 16). At the beginning, S2 stared at the computer screen and kept silent to the instructor's scaffolding (see Table 5: line 3, 5, 7, 9). Then, he was the first to understand the instructor's metacognitive scaffolding and successfully programmed to complete the task (see Table 5: line 10, 13, 15). S1 consistently kept silent after the scaffolding and looked at peer's computer operations in the subsequent learning (see Table 5: line 3, 5, 7, 9, 11, 12, 14, 16).

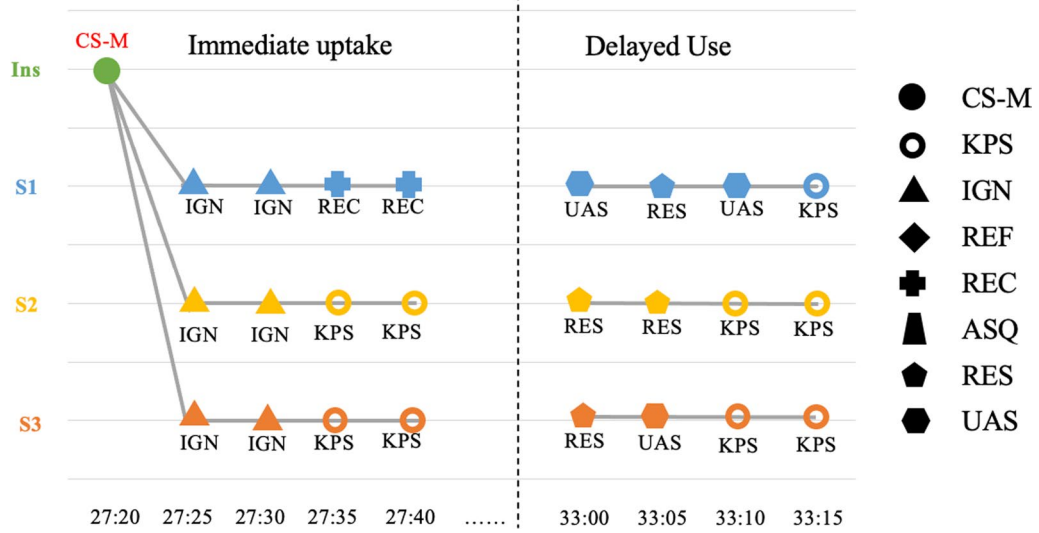

Fig. 9 The temporal transition after the CS-M scaffolding

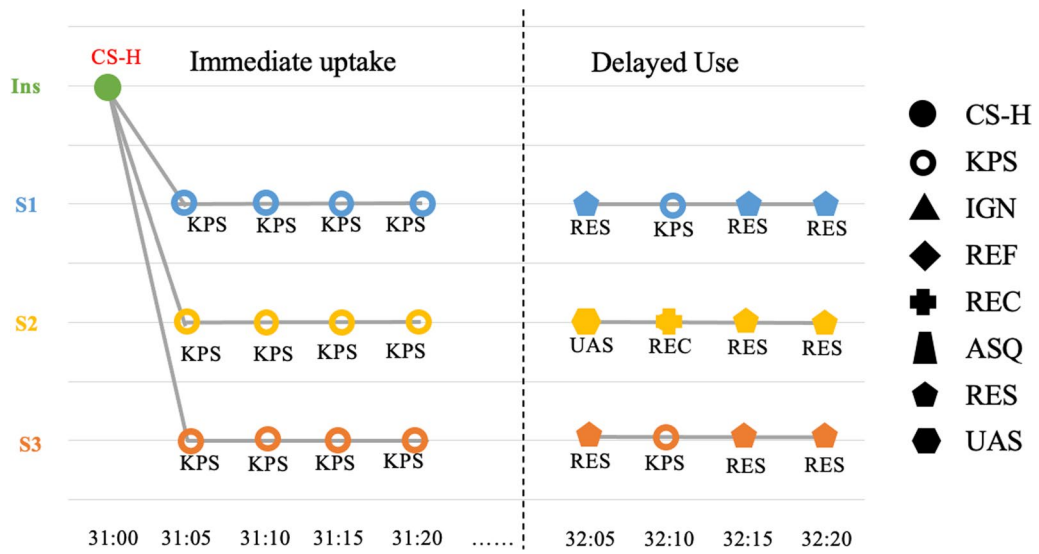

Fig. 10 The temporal transition after the CS-H scaffolding

In Group 7, the instructor prompted the group to initialize the programming before text recognition, and in subsequent collaborative learning (about five minutes after the instructor provided the scaffolding), the group initialized the programming after many debugging attempts without external assistance and finally succeeded in recognizing the idiom text. S1 was the most active student after the instructor provided the medium-control cognitive scaffolding (CS-M). He was the first to repeat the guidance from the instructor's scaffolding, which pointed out the steps for color recognition (see Table 6: line 4, 6). In the subsequent learning (i.e., the delayed use of scaffolding), S1 was also the first student to understand the instructor's scaffolding and solved problems with programming (see Table 6: line 4, 6, 8, 12). S2 ignored the scaffolding at the beginning and looked at the peer's computer operations with silences (see Table 6: line 2, 3, 5, 7). In the subsequent learning, he nodded and verbally showed agreement to the peer's opinion and silently looked at the computer screen (see Table 6: line 9, 11, 13). At the end, he brought up his understanding and pointed out the parameters that the group needed to

modify (see Table 6: line 14). S3 consistently kept silent and looked at peer's programming operations after the instructor offered medium-control cognitive scaffolding and in the subsequent learning (see Table 5: line 3, 5, 7, 9, 11, 12, 14, 16).

In Group 6, the instructor helped the group to set the parameters and complete the programming, and in subsequent collaborative learning (about five minutes after the instructor provided the scaffolding), without the instructor's assistance, the group successfully set the control parameters after several times of debugging. S1, S2 and S3 all kept silent and looked at the instructor's programming operations after instructor offered the high-control cognitive scaffolding (CS-H) about the parameter modification (see Table 7: line 2, 3, 4, 5). In the subsequent learning (i.e., the delayed use of scaffolding), S2 was the first to verbally show his understandings and use scaffolding to solve the problem through programming (see Table 7: line 6, 8). Then he nodded his head to show agreement to peer's opinions (see Table 7: line 11, 14). In the subsequent learning, S3 nodded her head to show agreement to peer's opinion and then verbally expressed her understanding of the scaffolding (see Table 7: line 7, 10). In the subsequent learning, S1 nodded, silently looked at peer's programming operations and verbally responded to peer's opinion (see Table 7: line 7, 9, 10, 12).

Discussion

Addressing research questions

This research utilized the multimodal learning analytics to analyze small groups' collaborative programming process, particularly examining the immediate and delayed effects of instructor scaffoldings on groups' collaborative programming. The results indicated that the instructor provided five types of scaffoldings from the social, cognitive (the superficial, medium, and high level), and metacognitive dimensions. Groups had seven types of responses (i.e., immediate uptakes and delayed uses) to these scaffoldings, ranging from the low to medium and high level of cognitive engagement. Five instructor scaffoldings had statistically significant differences on all seven codes of groups' responses, whether taking collaborative group state as a covariate or not. Groups in general had a high level of cognitive engagement after the instructor provided the low- and medium-control cognitive scaffoldings, while groups had a low level of cognitive engagement after the social scaffolding. Consistent with previous research (Barron, 2003; Maksic & Josic, 2021; van de Pol et al., 2019), the cognitive scaffoldings with low- and medium-level control were proved to be the most beneficial ones for improving students' cognitive engagement in collaborative programming.

Regarding the second research question, five types of instructor scaffoldings had different effects on the groups' social interactions and cognitive structures. Regarding the social interaction, groups in general had the highest frequency of instructor–student interaction and student–student interaction after the low- and medium-control cognitive scaffoldings. On the contrary, after the social scaffolding, groups had the lowest frequency of instructor–student interaction and student–student interaction. Comparing to previous research results (e.g., Maksic & Josic, 2021), our results showed that compared to the social scaffolding, cognitive scaffolding not only had positive effects on improving group cognition and problem-solving, but also increased students' social interaction and participation. Regarding the overall cognitive structure, groups in general had the most frequent connections with higher level of cognitive engagement after the metacognitive and low-control cognitive scaffoldings. On the contrary, after the social scaffolding, groups had the least connections regarding higher level of cognitive codes. Consistent with previous research results (Wittwer & Renkl, 2008; Wittwer et al., 2010), the instructor's regulation of groups' learning and low-level cognitive supports were beneficial for students to achieve a high-level engagement. Overall, the low/medium-control cognitive and metacognitive scaffoldings can improve students' social and cognitive engagement during the period of immediate uptakes; in contrast, the social scaffolding had limited positive effect in prompting group engagement.

Third, after the scaffolding faded, groups used the content from the high-control cognitive scaffolding frequently to solve problems in a delayed way, but groups did not use the instructor's scaffolding content from the social and low-control cognitive scaffoldings. Consistent with previous research results (Bulu & Pedersen, 2010; Tawfik et al., 2018), direct supports from the instructors' definite, complete information (such as higher control) had more delayed effect than the low-controlled scaffoldings. That is, students tended to use the contents, operations, or answers provided by the instructor's high level of cognitive supports in order to solve the programming problems after the scaffoldings faded (Margulieux & Catrambone, 2021; Tawfik et al., 2018). In addition, inconsistent with the findings of previous research (van de Pol et al., 2019), the groups' ignorance of the scaffoldings during the period of immediate uptakes did not negatively influence their delayed use of scaffoldings in subsequent collaborative learning. For example, there were four times groups showed a low-level cognitive engagement (such as kept silent, ignored or refused the scaffolding) when the instructor provided the scaffolding, but later they used the scaffolding content in a delayed

way to solve problems. Overall, the instructor's high-control cognitive scaffolding facilitated the use of content or hint within small groups in a delayed way to solve problems and complete collaborative tasks.

Pedagogical implications

The empirical research results provide three pedagogical implications to help instructors design and utilize scaffoldings to improve small groups' collaborative leaning. First, instructors should use the low- and medium-level of cognitive scaffoldings, along with metacognitive scaffoldings to initiate open-ended questions or provide hints to groups in order to improve collaborative programming quality. Consistent with previous research results (van de Pol et al., 2019; Wittwer et al., 2010), during collaborative programming, novice learners usually encounter complex tasks that cause high cognitive loads (Kunkle & Allen, 2016; Margulieux & Catrambone, 2021; Yeomans et al., 2019); in this case, the low- and medium-control cognitive scaffoldings can help students actively think about open-ended questions, process new information, and elaborate and reflect on ideas. With the cognitive supports, it is easier for groups to remove barriers caused by students' low cognitive levels and integrate new information and knowledge into their self-perception system (Wittwer & Renkl, 2008; Wittwer et al., 2010). In addition, instructors should make a conscious effort to provide metacognitive scaffolding, since the metacognitive support from the instructors can monitor and guide groups' learning pathways. This assistance can help students pay attention to knowledge-based programming tasks, and the cognitive support prompts students to think further on the existing learning materials or programming problems (Mattanah et al., 2005; van de Pol et al., 2019; Wittwer et al., 2010).

Second, strengthening the delayed effect of scaffoldings is conducive for groups to solve problems and construct knowledge by themselves in subsequent collaborative learning. The results showed that the high-control cognitive scaffolding helped groups understand and apply content or operation from the instructor and then solve collaborative problems in subsequent collaborative learning. Particularly, when students face problems of high difficult level, they need the instructor to provide direct assistance or offer answer to fill cognitive gaps in order to move on for problem-solving (Mattanah et al., 2005; Nedić et al., 2015). Consistent with previous research results (Bloome et al., 2009; Brown & Renshaw, 2009; Wiig et al., 2017), in some cases, although some student groups did not accept the scaffoldings immediately, but they understood and applied the content during the later stage and used it to solve programming problems after the scaffolding faded. Therefore, the instructors should consider the group collaborative states and characteristics and provide a high-control cognitive scaffolding when they face difficult problems, so as to facilitate groups' understanding and application of knowledge and complete collaborative learning tasks.

Finally, social scaffolding needs to be carefully considered to enhance immediate and delayed effects on group collaboration. The results showed that among five instructor scaffoldings, social scaffolding had the weakest positive effect on collaborative group learning, either on social interaction or cognitive engagement, which was different from previous research results. One of the reasons is that when groups faced programming problems beyond their cognitive levels, it might be difficult for them to communicate their own opinions with peers, and only when the cognitive barriers were cleared, they would be more motivated to share, discuss and reflect on ideas or problems (Wittwer & Renkl, 2008; Wittwer et al., 2010). Previous research indicated that social supports from the instructor or peers were conducive to build a supportive, safe atmosphere, which may encourage students' idea expressions and active participation in collaborative activities (Maksic & Josic, 2021; Ouyang et al., 2020). Taken together, we conclude that instructors should use social scaffolding carefully when students have a certain cognitive basis for the learning topics, in order to improve the scaffolding effect in collaborative learning.

Overall, the results suggest that instructors' support and scaffoldings are needed, particularly when learners lack sufficient understandings of content and hands-on experiences. The instructors should grasp and understand groups' difficulty levels as well as their cognitive load before providing scaffoldings, and then tailor the scaffolding to the groups' situation in order to provide appropriate supports (Meloth & Deering, 1999; Ouyang et al., 2021; Webb, 2009). To avoid premature fading of scaffolding effect, the instructor should check the group's learning state and make appropriate adjustments before leaving the group. Moreover, in eastern education system, such as China, instructors should also consider socio-cultural factors (Zhang, 2007) when providing the scaffoldings, in order to avoid students seeing the instructor as an authoritative source of knowledge and director of the learning process (Zhang, 2013). In summary, from the pedagogical perspective, instructors should consider scaffolding types, group states and characteristics, as well as the timing of scaffolding to better design and facilitate collaborative programming (Bliss et al., 1996; Ouyang et al., 2021; van de Pol et al., 2015).

Analytical implications

This research uses MMLA and multiple analytical methods to examine the immediate and delayed effects of instructor scaffoldings on the groups' collaborative programming. MMLA is a critical and significant breakthrough for analyzing collaborative programming processes, and previous studies have called for using MMLA in collaborative programming (Twiner et al., 2021). The CSCL community has recently called for collaborative learning analytics, the intersection of collaborative learning and learning analytics, to focus on analytic and implementation of learning analytics on groups' collaboration (Schneider et al., 2021). Because instructors and students communicate ideas and construct meanings through behaviors, discourses, gestures, and computer operations, it is necessary to collect and merge data from multiple sources in order to reflect the scattered activities of participants and alleviate the isolation of information perception caused by a single data source (Bezemer, 2008; Chatti et al., 2017). The multiple analytical methods are conducive to the analytics of collaborative learning from social, cognitive, behavioral, and micro-level dimensions, such that instructors can better understand and reflect on the process of collaborative learning, and use scaffoldings more skillfully to support collaborative learning (Maksic & Josic, 2021). Echoing this trend, this research collects and analyzes learners' in-class behaviors, computer operations and their communicative discourses synchronously, so as to conduct an integrated microanalysis of the moment-to-moment details of how learners coordinate their communications, behaviors, and movements during collaborative programming. Future studies can further synchronize audio discourse data, video recording data, facial expressions and eye tracking movements to better reveal the collaborative programming learning patterns (e.g., Chevalier et al., 2020; Sun & Hsu, 2019; Zatarain Cabada et al., 2018). Overall, an integration of MMLA and collaborative analytics can provide feedback on similarities and differences of CSCL results at the group level to improve the understanding of groups' collaborative work. An examination of instructor scaffoldings on collaborative learning can achieve a complete closed loop of analytics, feedback, and iterative application with a goal to develop collaborative learning quality and process.

Conclusion, limitations, and future direction

Collaborative programming is encouraged to be integrated in STEM education to transform education from the instructor-directed lecturing to the learner-centered learning. Using MMLA methods, this research provided a deep understanding of the immediate and delayed effects of instructor scaffoldings on small groups' collaborative programming in K-12 STEM education from a process-oriented perspective. The results indicate that various instructor scaffoldings have been used to promote groups' social and cognitive engagement. In addition, instructor scaffoldings have delayed effects on promoting collaborative programming qualities. Based on these findings, this research offers implications for pedagogical support and learning analytics of collaborative programming. There are three major limitations of this research. First, the research data are originated from the natural classroom environment, which lacks the deliberate design and control of the instructor scaffoldings. Future research should consider a quasi-experimental research design to examine and verify the immediate and delayed effects of different instructor scaffoldings. Second, this research is conducted with a small sample size in one class taught by one instructor; future research should apply scaffoldings to multiple courses or disciplines, expand the sample size and balance student gender, in order to further verify the results and implications. Third, future work should expand the multimodal data collections to include psychological and physical data, which can complement discourse and behavioral data, and better reveal the collaborative programming learning patterns. Overall, this research takes a step forward to conduct a holistic analysis of instructor scaffoldings as well as their effects on collaborative processes in computer programming education in China's K-12 education. Given the complexity of collaborative learning as well as the effects of instructor scaffoldings on collaborative learning, it is highly suggested that instructors should integrate scaffoldings to facilitate computer programming education and relevant research should combine multimodal data sources and multiple learning analytical methods to reveal details of the process of collaboration.

Acknowledgements

We appreciate the participation and support from the students, teachers and parents in this research.

Author contributions

FO designed the research, designed and facilitated data analysis and revised the manuscript; XD analyzed the data and wrote the first draft of the manuscript under FO's supervision; and SC collected data and contributed to the preliminary data analysis. All authors read and approved the final manuscript.

Funding

This work is supported by NSFC funding (62177041), Zhejiang Province educational science and planning research project (2022SCG256), and Zhejiang University graduate education research project (20220310).

Availability of data and materials

The data are available upon request from the corresponding author.

Declarations

Competing interests

The authors declare that they have no competing interests.

Received: 7 February 2022 Accepted: 30 June 2022
Published online: 08 July 2022

References

Barron, B. (2003). When smart groups fail. *Journal of the Learning Sciences, 12*(3), 307–359. https://doi.org/10.1207/S15327809JLS1203_1

Belland, B. R., Walker, A. E., Kim, N. J., & Lefler, M. (2017). Synthesizing results from empirical research on computer-based scaffolding in STEM education: A meta-analysis. *Review of Educational Research, 87*(2), 309–344. https://doi.org/10.3102/0034654316670999

Bezemer, J. (2008). Displaying orientation in the classroom: Students' multimodal responses to teacher instructions. *Linguistics and Education, 19*(2), 166–178. https://doi.org/10.1016/j.linged.2008.05.005

Bliss, J., Askew, M., & MacRae, S. (1996). Effective teaching and learning: Scaffolding revisited. *Oxford Review of Education, 22*(1), 37–61. https://doi.org/10.1080/0305498960220103

Bloome, D., Beierle, M., Grigorenko, M., & Goldman, S. (2009). Learning over time: Uses of intercontextuality, collective memories, and classroom chronotopes in the construction of learning opportunities in a ninth-grade language arts classroom. *Language and Education, 23*(4), 313–334. https://doi.org/10.1080/09500780902954257

Brown, R., & Renshaw, P. (2009). Positioning students as actors and authors: A chronotopic analysis of collaborative learning activities. *Mind, Culture and Activity, 13*(3), 247–259. https://doi.org/10.1207/s15327884mca1303_6

Bulu, S. T., & Pedersen, S. (2010). Scaffolding middle school students' content knowledge and ill-structured problem solving in a problem-based hypermedia learning environment. *Educational Technology Research & Development, 58*(5), 507–529. https://doi.org/10.1007/s11423-010-9150-9

Chatti, M. A., Muslim, A., & Schroeder, U. (2017). Toward an open learning analytics ecosystem. In B. K. Daniel (Ed.), *Big data and learning analytics in higher education* (pp. 195–219). Cham: Springer. https://doi.org/10.1007/978-3-319-06520-5_12

Chevalier, M., Giang, C., Piatti, A., & Mondada, F. (2020). Fostering computational thinking through educational robotics: A model for creative computational problem solving. *International Journal of STEM Education, 7*(1), 1–18. https://doi.org/10.1186/s40594-020-00238-z

Chinese Ministry of Education. (2019). *Report on the development of online learning in China 2019*. Tsinghua University Press.

Chiu, T. K. F. (2021). A holistic approach to the design of artificial intelligence (AI) education for K-12 schools. *TechTrends, 65*(5), 796–807. https://doi.org/10.1007/s11528-021-00637-1

Clark, D. B., & Sengupta, P. (2020). Reconceptualizing games for integrating computational thinking and science as practice: Collaborative agent-based disciplinarily-integrated games. *Interactive Learning Environments, 28*(3), 328–346. https://doi.org/10.1080/10494820.2019.1636071

Cress, U., Rosé, C., Wise, A. F., & Oshima, J. (2021). International handbook of computer-supported collaborative learning. *Springer*. https://doi.org/10.1007/978-3-030-65291-3

Damşa, C. I., & Nerland, M. (2016). Student learning through participation in inquiry activities: Two case studies in teacher and computer engineering education. *Vocations and Learning, 9*(3), 275–294. https://doi.org/10.1007/s12186-016-9152-9

Demir, O., & Seferoglu, S. S. (2020a). A comparison of solo and pair programming in terms of flow experience, coding quality, and coding achievement. *Journal of Educational Computing Research, 58*(8), 1448–1466. https://doi.org/10.1177/0735633120949788

Demir, O., & Seferoglu, S. S. (2020b). The effect of determining pair programming groups according to various individual difference variables on group compatibility, flow, and coding performance. *Journal of Educational Computing Research, 59*(1), 41–70. https://doi.org/10.1177/0735633120949787

Di Mitri, D., Schneider, J., & Drachsler, H. (2021). Keep me in the loop: Real-time feedback with multimodal data. *International Journal of Artificial Intelligence in Education*. https://doi.org/10.1007/s40593-021-00281-z

Dillenbourg, P. (1999). What do you mean by "collaborative learning"? In P. Dillenbourg (Ed.), *Collaborative learning: Cognitive and computational approaches* (Vol. 1, pp. 1–15). Elsevier.

Fanchamps, L. J. A., Slangen, L., Hennissen, P., & Specht, M. M. (2021). The influence of SRA programming on algorithmic thinking and self-efficacy using Lego robotics in two types of instruction. *International Journal of Technology and Design Education, 31*(2), 203–222. https://doi.org/10.1007/s10798-019-09559-9

Fedorenko, E., Ivanova, A., Dhamala, R., & Bers, M. U. (2019). The language of programming: A cognitive perspective. *Trends in Cognitive Sciences, 23*(7), 525–528. https://doi.org/10.1016/j.tics.2019.04.010

Gaul, C., & Kim, M. K. (2020). Learner participation regulation supported by long-term peer moderation and participation feedback during asynchronous discussions. *Journal of Computers in Education, 7*(3), 295–331. https://doi.org/10.1007/s40692-020-00158-5

Gidalevich, S., & Kramarski, B. (2019). The value of fixed versus faded self-regulatory scaffolds on fourth graders' mathematical problem solving. *Instructional Science, 47*(1), 39–68. https://doi.org/10.1007/s11251-018-9475-z

Goodyear, P., Jones, C., & Thomson, K. (2014). Computer-supported collaborative learning: Instructional approaches, group processes and educational designs. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of research on educational communications and technology* (pp. 439–451). Springer.

Gresalfi, M., Martin, T., Hand, V., & Greeno, J. (2009). Constructing competence: An analysis of student participation in the activity systems of mathematics classrooms. *Educational Studies in Mathematics, 70*(1), 49–70. https://doi.org/10.1515/ctra-2016-0019

Grévisse, C., Rothkugel, S., & Reuter, R. A. P. (2019). Scaffolding support through integration of learning material. *Smart Learning Environments, 6*(1), 1–24. https://doi.org/10.1186/s40561-019-0107-0

Grossen, M., & Bachmann, K. (2000). Learning to collaborate in a peer-tutoring situation: Who learns? What is learned? *European Journal of Psychology of Education, 15*(4), 491–508. https://doi.org/10.1007/BF03172990

Guzdial, M., & Morrison, B. (2016). Growing computer science education into a STEM education discipline. *Communications of the ACM, 59*(11), 31–33. https://doi.org/10.1145/3000612

Holmes, N. G., Day, J., Park, A. H. K., Bonn, D. A., & Roll, I. (2014). Making the failure more productive: Scaffolding the invention process to improve inquiry behaviors and outcomes in invention activities. *Instructional Science, 42*(4), 523–538. https://doi.org/10.1007/s11251-013-9300-7

Hmelo-Silver, C. E., & DeSimone, C. (2013). Problem-based learning: An instructional model of collaborative learning. In C. E. Hmelo-Silver, C. A. Chinn, C. K. K. Chan, & A. M. O'Donnell (Eds.), *The international handbook of collaborative learning* (1st ed., pp. 370–385). New York, NY: Routledge.

Hwang, W. Y., Shadiev, R., Wang, C. Y., & Huang, Z. H. (2012). A pilot study of cooperative programming learning behavior and its relationship with students' learning performance. *Computers & Education, 58*(4), 1267–1281. https://doi.org/10.1016/j.compedu.2011.12.009

Jadallah, M., Anderson, R. C., Nguyen-Jahiel, K., Miller, B. W., Kim, I. H., Kuo, L. J., Dong, T., & Wu, X. (2011). Influence of a teacher's scaffolding moves during child-led small group discussions. *American Educational Research Journal, 48*(1), 194–230. https://doi.org/10.3102/0002831210371498

Johnson, E. K. (2019). Waves: Scaffolding self-regulated learning to teach science in a whole-body educational game. *Journal of Science Education and Technology, 28*(2), 133–151. https://doi.org/10.1007/s10956-018-9753-1

Khan, S. M. (2017). Multimodal behavioral analytics in intelligent learning and assessment systems. *Innovative assessment of collaboration* (pp. 173–184). Springer. https://doi.org/10.1007/978-3-319-33261-1_11

Kiemer, K., Gröschner, A., Pehmer, A. K., & Seidel, T. (2015). Effects of a classroom discourse intervention on teachers' practice and students' motivation to learn mathematics and science. *Learning and Instruction, 35*, 94–103. https://doi.org/10.1016/j.learninstruc.2014.10.003

Kim, N. J., Vicentini, C. R., & Belland, B. R. (2022). Influence of scaffolding on information literacy and argumentation skills in virtual field trips and problem-based learning for scientific problem solving. *International Journal of Science and Mathematics Education, 20*(2), 215–236. https://doi.org/10.1007/s10763-020-10145-y

Kirschner, P., Sweller, J., & Clark, R. (2006). Why unguided learning does not work: An analysis of the failure of discovery learning, problem-based learning, experiential learning and inquiry-based learning. *Educational Psychologist*, *41*(2), 75–86. https://doi.org/10.1207/s15326985ep4102_1.

Kunkle, W. M., & Allen, R. B. (2016). The impact of different teaching approaches and languages on student learning of introductory programming concepts. *ACM Transactions on Computing Education, 16*(1), 1–26. https://doi.org/10.1145/2785807

Kwon, K., Hong, R., & Laffey, J. M. (2013). The educational impact of metacognitive group coordination in computer-supported collaborative learning. *Computers in Human Behavior, 29*(4), 1271–1281. https://doi.org/10.1016/j.chb.2013.01.003

Kynigos, C., & Diamantidis, D. (2022). Creativity in engineering mathematical models through programming. *ZDM-Mathematics Education, 54*, 149–162. https://doi.org/10.1007/s11858-021-01314-6

Lai, X., & Wong, G. K. W. (2022). Collaborative versus individual problem solving in computational thinking through programming: A meta-analysis. *British Journal of Educational Technology, 53*(1), 150–170. https://doi.org/10.1111/bjet.13157

Lavonen, J., Meisalo, V., & Lattu, M. (2002). Collaborative problem solving in a control technology learning environment, a pilot study. *International Journal of Technology and Design Education, 12*(2), 139–160. https://doi.org/10.1023/A:1015261004362

Lewis, C. (2012). The importance of students' attention to program state: A case study of debugging behavior. In C. Alison, S. Kate, & S. Beth (Eds.), *Proceedings of the 9th annual international conference on international computing education research* (pp. 127–134). Association for Computing Machinery.

Lin, Y., Wu, C., & Chiu, C. (2018). The use of wiki in teaching programming: Effects upon achievement, attitudes, and collaborative programming behaviors. *International Journal of Distance Education Technologies, 16*(3), 18–45. https://doi.org/10.4018/IJDET.2018070102

Maksic, S., & Josic, S. (2021). Scaffolding the development of creativity from the students' perspective. *Thinking Skills and Creativity, 41*, 100835. https://doi.org/10.1016/j.tsc.2021.100835

Margulieux, L. E., & Catrambone, R. (2021). Scaffolding problem solving with learners' own self explanations of subgoals. *Journal of Computing in Higher Education, 33*(2), 499–523. https://doi.org/10.1007/s12528-021-09275-1

Mattanah, J. F., Pratt, W. P., Cowan, P. A., & Cowan, C. P. (2005). Authoritative parenting, parental scaffolding of long-division mathematics, and children's academic competence in fourth grade. *Journal of Applied Developmental Psychology*, *26*(1), 85–106. https://doi.org/10.1016/j.appdev.2004.10.007.

Meloth, M. S., & Deering, P. D. (1999). The role of the teacher in promoting cognitive processing during collaborative learning. In A. M. O'Donnell & A. King (Eds.), *Cognitive perspectives on peer learning* (pp. 235–255). Lawrence Erlbaum Associates Publishers.

Nedić, J., Jošić, S., & Baucal, A. (2015). The role of asymmetrical interaction in the assessment of nonverbal abilities of children from the drop-in center. *Teaching Innovations, 28*(3), 189–206. https://doi.org/10.5937/inovacije1503189N

Ouyang, F., Chang, Y. H., Scharber, C., Jiao, P., & Huang, T. (2020). Examining the instructor-student collaborative partnership in an online learning community course. *Instructional Science, 48*(2), 183–204. https://doi.org/10.1007/s11251-020-09507-4

Ouyang, F., Chen, S., Yang, Y., & Chen, Y. (2022). Examining the effects of three group-level metacognitive scaffoldings on in-service teachers' knowledge building. *Journal of Educational Computing Research, 60*(2), 352–379. https://doi.org/10.1177/07356331211030847

Ouyang, F., Chen, Z., Cheng, M., Tang, Z., & Su, C.-Y. (2021). Exploring the effect of three scaffoldings on the collaborative problem-solving processes in China's higher education. *International Journal of Educational Technology in Higher Education, 18*(35), 1–22. https://doi.org/10.1186/s41239-021-00273-y

Ouyang, F., & Xu, W. (2022). The effects of three instructor participatory roles on a small group's collaborative concept mapping. *Journal of Educational Computing Research, 60*(4), 930–959. https://doi.org/10.1177/07356331211057283

Papadakis, S. (2018). Is pair programming more effective than solo programming for secondary education novice programmers? A case study. *International Journal of Web-Based Learning and Teaching Technologies, 13*(1), 1–16. https://doi.org/10.1177/1555412015622345

Pedró, F., Subosa, M., Rivas, A., & Valverde, P. (2019). *Artificial intelligence in education: Challenges and opportunities for sustainable development*. United Nations Educational, Scientific and Cultural Organization.

Perret-Clermont, A. N. (1980). *Social interaction and cognitive development in children*. Academic Press.

Plonka, L., Sharp, H., Van der Linden, J., & Dittrich, Y. (2015). Knowledge transfer in pair programming: An in-depth analysis. *International Journal of Human–computer Studies, 73*, 66–78. https://doi.org/10.1016/j.ijhcs.2014.09.001

Rasku-Puttonen, H., Eteläpelto, A., Arvaja, M., & Häkkinen, P. (2003). Is successful scaffolding an illusion?—Shifting patterns of responsibility and control in teacher–student interaction during a long-term learning project. *Instructional Science, 31*(6), 377–393. https://doi.org/10.1023/A:1025700810376

Riikonen, S., Seitamaa-Hakkarainen, P., & Hakkarainen, K. (2020). Bringing maker practices to school: Tracing discursive and materially mediated aspects of student teams' collaborative making processes. *International Journal of Computer-Supported Collaborative Learning, 15*(3), 319–349. https://doi.org/10.1007/s11412-020-09330-6

Roehler, L. R., & Cantlon, D. J. (1997). Scaffolding: A powerful tool in social constructivist classrooms. In K. Hogan & M. Pressley (Eds.), *Scaffolding student learning: Instructional approaches and issues* (pp. 6–42). Brookline.

Roschelle, J., & Teasley, S. (1995). The construction of shared knowledge in collaborative problem solving. In C. O'Malley (Ed.), *Computer-supported collaborative learning* (pp. 69–197). Springer.

Samuelsen, J., Chen, W., & Wasson, B. (2019). Integrating multiple data sources for learning analytics—Review of literature. *Research and Practice in Technology Enhanced Learning, 14*(1), 1–20. https://doi.org/10.1186/s41039-019-0105-4

Sandoval, W. A., & Reiser, B. J. (2004). Explanation-driven inquiry: Integrating conceptual and epistemic scaffolds for scientific inquiry. *Science Education, 88*(3), 345–372. https://doi.org/10.1002/sce.10130

Schneider, B., Dowell, N., & Thompson, K. (2021). Collaboration analytics—current state and potential futures. *Journal of Learning Analytics, 8*(1), 1–12.

Serrano-Cámara, L. M., Paredes-Velasco, M., Alcover, C. M., & Velazquez-Iturbide, J. Á. (2014). An evaluation of students' motivation in computer-supported collaborative learning of programming concepts. *Computers in Human Behavior, 31*, 499–508. https://doi.org/10.1016/j.chb.2013.04.030

Shaffer, D. W., Collier, W., & Ruis, A. R. (2016). A tutorial on epistemic network analysis: Analyzing the structure of connections in cognitive, social, and interaction data. *Journal of Learning Analytics, 3*(3), 9–45. https://doi.org/10.18608/jla.2016.33.3

Shaffer, D. W., Hatfield, D. L., Svarovsky, G. N., Nash, P., Nulty, A., Bagley, E., Frank, K. A., Rupp, A. A., & Mislevy, R. (2009). Epistemic network analysis: A prototype for 21st century assessment of learning. *International Journal of Learning and Media, 1*(2), 1–21. https://doi.org/10.1162/ijlm.2009.0013

Shaffer, D. W., & Ruis, A. R. (2017). Epistemic network analysis: A worked example of theory-based learning analytics. In C. Lang, G. Siemens, A. Wise, & D. Gašević (Eds.), *Handbook of learning analytics* (pp. 175–187). Society for Learning Analytics and Research.

Sinha, S., Rogat, T. K., Adams-Wiggins, K. R., & Hmelo-Silver, C. E. (2015). Collaborative group engagement in a computer-supported inquiry learning environment. *International Journal of Computer-Supported Collaborative Learning, 10*(3), 273–307. https://doi.org/10.1007/s11412-015-9218-y

Socratous, C., & Ioannou, A. (2021). Structured or unstructured educational robotics curriculum? A study of debugging in block-based programming. *Educational Technology Research and Development, 69*(6), 3081–3100. https://doi.org/10.1007/s11423-021-10056-x

Socratous, C., & Ioannou, A. (2022). Evaluating the impact of the curriculum structure on group metacognition during collaborative problem-solving using educational robotics. *TechTrends*. https://doi.org/10.1007/s11528-022-00738-5

Stahl, G. (2009). *Studying virtual math teams*. New York: Springer. Available at http://GerryStahl.net/vmt/book.

Sun, D., Ouyang, F., Li, Y., & Zhu, C. (2021). Comparing learners' knowledge, behaviors, and attitudes between two instructional modes of computer programming in secondary education. *International Journal of STEM Education, 8*(54), 1–15. https://doi.org/10.1186/s40594-021-00311-1

Sun, J. C., & Hsu, K. Y. (2019). A smart eye-tracking feedback scaffolding approach to improving students' learning self-efficacy and performance in a C programming course. *Computers in Human Behavior, 95*, 66–72. https://doi.org/10.1016/j.chb.2019.01.036

Tartas, V., & Perret-Clermont, A. N. (2008). Socio-cognitive dynamics in dyadic interaction: How do you work together to solve Kohs cubes? *European Journal of Developmental Psychology, 5*(5), 561–584. https://doi.org/10.1080/17405620701859522

Tawfik, A. A., Law, V., Ge, X., Xing, W., & Kim, K. (2018). The effect of sustained vs. faded scaffolding on students' argumentation in ill-structured problem solving. *Computers in Human Behavior, 87*, 436–449. https://doi.org/10.1016/j.chb.2018.01.035

Teague, D., & Roe, P. (2008). Collaborative learning—Towards a solution for novice programmers. In S. Hamilton & M. Hamilton (Eds.), *Proceedings of the tenth conference on Australasian computing education—CRPIT Volume 78* (pp. 147–153). Australian Computer Society, Australia.

Tohyama, S., Matsuzawa, Y., Yokoyama, S., Koguchi, T., & Takeuchi, Y. (2018). Constructive interaction on collaborative programming: Case study for grade 6 students group. *IFIP world conference on computers in education* (pp. 589–598). Springer International Publishing. https://doi.org/10.1007/978-3-319-74310-3_59

Twiner, A., Littleton, K., Whitelock, D., & Coffin, C. (2021). Combining socio-cultural discourse analysis and multimodal analysis to explore teachers' and pupils' meaning making. *Learning, Culture and Social Interaction, 30*, 100520. https://doi.org/10.1016/j.lcsi.2021.100520

Umapathy, K., & Ritzhaupt, A. D. (2017). A meta-analysis of pair-programming in computer programming courses: Implications for educational practice. *ACM Transactions on Computing Education, 17*(4), 1–13. https://doi.org/10.1145/2996201

van de Pol, J., Mercer, N., & Volman, M. (2019). Scaffolding student understanding in small-group work: Students' uptake of teacher support in subsequent small-group interaction. *Journal of the Learning Sciences, 28*(2), 206–239. https://doi.org/10.1080/10508406.2018.1522258

van de Pol, J., Volman, M., & Beishuizen, J. (2010). Scaffolding in teacher–student interaction: A decade of research. *Educational Psychology Review, 22*(3), 271–296. https://doi.org/10.1007/s10648-010-9127-6

van de Pol, J., Volman, M., Oort, F., & Beishuizen, J. (2015). The effects of scaffolding in the classroom: Support contingency and student independent working time in relation to student achievement, task effort and appreciation of support. *Instructional Science, 43*(5), 615–641. https://doi.org/10.1007/s11251-015-9351-z

Veerasamy, A. K., D'Souza, D., Linden, R., & Laakso, M. J. (2019). Relationship between perceived problem-solving skills and academic performance of novice learners in introductory programming courses. *Journal of Computer Assisted Learning, 35*, 246–255. https://doi.org/10.1111/jcal.12326

Villamor, M. M., & Rodrigo, M. M. T. (2019). Gaze collaboration patterns of successful and unsuccessful programming pairs using cross-recurrence quantification analysis. *Research and Practice in Technology Enhanced Learning, 14*(1), 1–22. https://doi.org/10.1186/s41039-019-0118-z

Wang, X., & Hwang, G. (2017). A problem posing-based practicing strategy for facilitating students' computer programming skills in the team-based learning mode. *Educational Technology Research and Development, 65*(6), 1655–1671. https://doi.org/10.1007/s11423-017-9551-0

Webb, N. M. (2009). The teacher's role in promoting collaborative dialogue in the classroom. *British Journal of Educational Psychology, 79*, 1–28. https://doi.org/10.1348/000709908X380772

Webb, N. M., & Farivar, S. (1999). Developing productive group interaction in middle school. In A. M. O'Donnell & A. King (Eds.), *Cognitive perspectives on peer learning* (pp. 117–149). Lawrence Erlbaum Associates.

Webb, N. M., Nemer, K. M., & Ing, M. (2006). Small group reflections: Parallels between teacher discourse and student behavior in peer-directed groups. *Journal of the Learning Sciences, 15*(1), 63–119. https://doi.org/10.1207/s15327809jls1501_8

Wei, X., Lin, L., Meng, N., Tan, W., & Kong, S. C. (2020). The effectiveness of partial pair programming on elementary school students' computational thinking skills and self-efficacy. *Computers & Education, 160*, 104023. https://doi.org/10.1016/j.compedu.2020.104023

Wiig, C., Silseth, K., & Erstad, O. (2017). Creating intercontextuality in students learning trajectories. Opportunities and difficulties. *Language and Education, 32*(1), 43–59. https://doi.org/10.1080/09500782.2017.1367799

Wittwer, J., Nückles, M., Landmann, N., & Renkl, A. (2010). Can tutors be supported in giving effective explanations? *Journal of Educational Psychology, 102*(1), 74–89. https://doi.org/10.1037/a0016727

Wittwer, J., & Renkl, A. (2008). Why instructional explanations often do not work: A framework for understanding the effectiveness of instruction explanations. *Educational Psychologist, 43*(1), 49–64. https://doi.org/10.1080/00461520701756420

Wu, B., Hu, Y., Ruis, A. R., & Wang, M. (2019). Analysing computational thinking in collaborative programming: A quantitative ethnography approach. *Journal of Computer Assisted Learning, 35*(3), 421–434. https://doi.org/10.1111/jcal.12348

Yeomans, L., Zschaler, S., & Coate, K. (2019). Transformative and troublesome? Students' and professional programmers' perspectives on difficult concepts in programming. *ACM Transactions on Computing Education, 19*(3), 1–27. https://doi.org/10.1145/3283071

Zatarain Cabada, R., Barrón Estrada, M. L., Ríos Félix, J. M., & Alor Hernández, G. (2018). A virtual environment for learning computer coding using gamification and emotion recognition. *Interactive Learning Environments, 28*(8), 1048–1063. https://doi.org/10.1080/10494820.2018.1558256

Zhang, J. (2007). A cultural look at information and communication technologies in Eastern education. *Educational Technology Research and Development, 55*(3), 301–314. https://doi.org/10.1007/s11423-007-9040-y

Zhang, J. (2013). Chapter 28: Collaboration, technology, and culture. In C. E. Hmelo-Silver, C. A. Chinn, C. K. K. Chan, & A. M. O'Donnell (Eds.), *The international handbook of collaborative learning* (pp. 495–508). Routledge.

Zheng, L. (2021). Improving programming skills through an innovative collaborative programming model: A case study. Lecture notes in educational technology*Data-driven design for computer-supported collaborative learning* (pp. 75–85). Springer. https://doi.org/10.1007/978-981-16-1718-8_6

Zheng, L., Zhen, Y., Niu, J., & Zhong, L. (2022). An exploratory study on fade-in versus fade-out scaffolding for novice programmers in online collaborative programming settings. *Journal of Computing in Higher Education*. https://doi.org/10.1007/s12528-021-09307-w

Publisher's Note