Computers & Education 58 (2012) 1267–1281

Contents lists available at SciVerse ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

A pilot study of cooperative programming learning behavior and its relationship with students' learning performance

Wu-Yuin Hwang [a], Rustam Shadiev [a], Chin-Yu Wang [b,*], Zhi-Hua Huang [a]

[a] Graduate Institute of Network Learning Technology, National Central University, 300 Jhongda Road, Jhongli City, Taoyuan County 32001, Taiwan, ROC
[b] Department of Tourism, Providence University, 200 Chungchi Road, Shalu, Taichung County 43301, Taiwan, ROC

ARTICLE INFO

Article history:
Received 26 August 2011
Received in revised form
16 November 2011
Accepted 13 December 2011

Keywords:
Cooperative/collaborative learning
Distributed learning environments
Interactive learning environments
Pedagogical issues
Programming and programming languages

ABSTRACT

In this study we proposed a web-based programming assisted system for cooperation (WPASC) and we also designed one learning activity for facilitating students' cooperative programming learning. The aim of this study was to investigate cooperative programming learning behavior of students and its relationship with learning performance. Students' opinions and perceptions toward learning activity and the WPASC were also investigated. The results of this study revealed that most of students perceived that learning activity and the WPASC were useful for cooperative programming learning. Students' learning behavior during cooperative programming learning activity was classified into six different categories and we found that learning behavior has relationship with learning performance. Students from *completely independent, self-improving using assistance, confident after enlightenment* and *imitating* categories performed well due to their effective and motivated learning behavior. However, students from *performing poorly without assistance* and *plagiarizing* categories performed the worse; the former could not get assistance at all and the later had no learning motivation. The results also showed that students' learning behavior may have *increasing*, *decreasing* and *no transition* during problems solving. Therefore, *performing poorly without assistance* and *plagiarizing* learning behavior and *decreasing transition* or *no transition* in learning behavior should be identified right after completing a programming problem. Then the instructor should intervene into learning behavior in order to change it into more effective for learning. Besides, more incentives need to be given for increasing students' learning motivation and posting solutions and feedback by students at the early stage of a problem solving period.

1. Introduction

The rapid development of information technology created high demand of knowledgeable and skillful programming specialists in a market. Therefore, designing and delivering an appropriate instruction supported by programming learning systems for preparing students with such characteristics efficiently defined as an important task nowadays. Most traditional programming instructions usually take place in a computer classroom where an instructor focus on syntax, logics, concepts, and analysis of program codes through lecturing and discussion (Bouton & Garth, 1983; Sharan, 1980; Slavin, 1995). Such method of instruction limits learning effectiveness as students have limited opportunities to practice programming skills and instructors cannot be sure if the learning context suits each student. Moreover, learning programming is not easy for many students, especially novices and those without relevant background. Shen and Sun (2000), Bravo, Marcelino, Gomes, Esteves, and Mendes (2005) and McDowell, Werner, Bullock, and Fernald (2002) therefore suggested integration of cooperative programming learning activities into instructional design. Chiu (2008) argued cooperative learning offers many potential benefits beyond programming learning, i.e., students capitalize on one another's resources and skills (asking one another for information, evaluating one another's ideas, monitoring one another's work, etc.). Several researches were carried out on exploring students' learning behavior during cooperative programming (Bravo et al., 2005; Hwang, Wang, Hwang, Huang, & Huang, 2008; McDowell et al., 2002; Webb, Nemer, & Ing, 2006; Williams & Upchurch, 2001). However, Preston (2005) noticed that exploring students' programming learning behavior engaged in an individual task with and without assistance from others (Sandholtz, 2000; Webb et al., 2006), its classification and relationship with learning achievement does not appear to be common practice in the programming pedagogy.

* Corresponding author. Tel.: +886 4 26328001x13519; fax: +886 4 26530035.
E-mail address: wang@pu.edu.tw (C.-Y. Wang).

0360-1315/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compedu.2011.12.009

Bravo et al. (2005), Butz, Hua, and Maguire (2004), Hwang et al. (2008), Kersten and Murphy (1999), Mosconi, Ottelli, and Porta (2003) and Shen and Sun (2000) developed and proposed web-based programming learning systems to support students' programming learning. Hwang et al. (2008) proposed writing source codes to solve problems, program gap filling, execution of program, debugging practice and peer feedback activities for effective programming learning and improving programming skills. However, detailed review of current web-based programming learning systems showed that none of them could support all of learning activities.

Student's learning behavior during cooperative programming learning activities can be identified then be classified into several categories. For example, student who absorbs and applies new knowledge into practice without any assistance can be classified into category of *completely independent*; student who is assisted before accomplishing an assignment and then provide assistance such as sharing source codes and feedback with peers can be classified into category of *confident after enlightenment*; student who has insufficient programming skills and no confidence to complete assignment on their own can also be identified. Moreover, student's learning behavior may change from one to another during cooperative programming learning activities.

In this study we proposed a web-based programming assisted system for cooperation (WPASC) and we designed learning activity for facilitating students' cooperative programming learning activity. Students could write source codes, fill in the gap, execute and debug source codes with support of WPASC. We also encouraged students to seek for assistance and provide assistance to peers. The aim of this study was to investigate cooperative programming learning behavior of students and its relationship with learning performance. Students' opinions and perceptions toward learning activity and the WPASC were also investigated.

The remainder of the paper is organized as follows. First, literature regarding programming learning and cooperation and current web-based programming learning systems are reviewed. Then, underlying method of the study is presented. The results and pedagogical implications of the study are then discussed. Finally, a few concluding remarks are given.

2. Literature review

2.1. Programming learning and cooperation

Nilson (2010) argued that lecturing become the most efficient method of instruction if it is used for right purposes, it is carefully prepared and eloquently delivered, and it is supplemented with thought-provoking student learning activities. Truong, Bancroft, and Roe (2003) suggested programming problem solving as a potential learning activity that allows students practice programming skills over and over for the purpose of improving or mastering it. According to Hwang et al. (2008), programming problem solving includes writing source codes, gap filling, execution of program, debugging practice and peer feedback activities. For programming learning to be more effective, Hwang et al. suggested writing codes should be organized as multiple activities in proper sequence, from simple to complex. Using "fill in the gap" programming exercises helps novice programmers access prior knowledge, and knowledge learnt during the class and apply it to a new problem. According to Lee and Wu (1999), debugging of program practice is effective in improving novice programmers' programming skills. As for peer feedback, it was considered as effective activity in promoting students' higher cognitive skills since students use their knowledge and skills to interpret, analyze and evaluate others' work in order to clarify and correct it (Hattie & Timperley, 2007; Hwang et al., 2008; Walker, Rummel, & Koedinger, 2011). Kolb (1984) conceived learning as a four stages cycle composed of concrete experience, reflective observation, abstract conceptualization, and active experimentation. Concrete experience was suggested as a good starter for students' learning process. That is, in programming learning courses, practice is important for improving students' learning. Students should be given enough practice opportunities in an environment where they can receive constructive and corrective feedback (Ben-Ari, 2001). If students practice frequently then their programming skills may improve (Truong et al., 2003).

Byrne and Lyons (2001) followed Kolb's experiential learning theory and classified programming learners into converging, diverging, accommodating, and assimilating learning styles. "Convergers" learn better when provided with practical applications of concepts and theories, they can integrate a number of different concepts to produce a complete set of program codes. "Divergers" learn better when allowed to observe and collect a wide range of information, they are able to extract a portion of a set of program codes and use it for other purposes. "Accommodators" learn better when provided with "hands-on" experiences, they are capable of debugging program codes to improve the quality of the program. "Assimilators" learn better when presented with sound logical theories to consider, they tend to analyze others' program codes to enhance the quality of their codes. Byrne and Lyons found that "convergers" performed best overall and they argued that "convergers" combine abstract conceptualization and active experimentation, and are deemed best at finding practical uses for ideas and theories. Their strengths are said to be in problem solving, decision-making, deductive reasoning and defining problems. This combines many of the attributes which are required for successful programmers.

Bouton and Garth (1983), Hwang et al. (2008), Sharan (1980), and Slavin (1995) suggested programming learning in cooperation as it offers many potential benefits beyond programming learning. For example, it motivates students programming learning and active participation. During cooperative programming learning students divide up tasks, work together, support each other, learn from each other, and share experiences to achieve learning objectives. Preston (2005) cited the importance of providing each student with some time to work with his or her partner and lab sessions should be part of the course. Besides, Bouton and Garth (1983), Hwang et al. (2008), Sharan (1980), and Slavin (1995) argued that cooperative programming learning promote exchange of ideas among learners and also allow learners to develop better learning processes, experiences, and outcomes comparing to learning in traditional contexts. Nosek (1998), McDowell et al. (2002), and Williams and Upchurch (2001) found that students working in cooperation will spend less time to solve a programming problem and solve it better than if they would working alone. Bravo et al. (2005) suggested that cooperative programming learning can facilitate cognitive development in learners and also increase their learning motivation.

The literature on cooperative learning defined cooperative behavior as students working on assigned problems together, discussing them and correcting any misconceptions or mistakes (Davidson, 1994). Smith (1995) analyzed students working in groups and he identified several types of learning behavior. Two of them were traditional classroom learning group, and cooperative learning group. He characterized students in the traditional classroom learning group as caring only about their part of the group assignment. These students believe they will be evaluated and rewarded as individuals. Therefore these students interact primarily to clarify how assignments are to be done, they have no motivation to offer assistance to others, and some of them even seek free ride on the efforts of others. Students in the cooperative