Computers in Human Behavior 64 (2016) 423–431

Contents lists available at ScienceDirect

Computers in Human Behavior

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

Full length article

The impact of social factors on pair programming in a primary school

Baichang Zhong [a, *], Qiyun Wang [b], Jie Chen [c]

[a] Collaborative Innovation Center for Talent Cultivating Mode in Basic Education, Nanjing Normal University, China
[b] National Institute of Education, Nanyang Technological University, Singapore
[c] Bixi Primary School of Changshu City, China

ARTICLE INFO

Article history:
Received 19 March 2016
Accepted 18 July 2016

Keywords:
Primary education
Pair programming
Social factors
Gender
Partnership

ABSTRACT

Pair programming (PP) is a usefulness approach to fostering computational thinking (CT) for young students. However, there are many factors to impact the effectiveness of PP. Among all factors, the social factors are often ignored by researchers. Therefore, this study aimed to explore the impact of two social factors (gender and partnership) on PP in a primary school setting. To that end, we conducted PP experiments in four classes from the sixth grade in a Chinese primary school. The research results indicated: (a) there was no significant difference on compatibility among the gender pairs, but a significant difference among partnership pairs; (b) there was no significant difference on programming achievement and confidence among different pairs, and girls became more productive and confidence in PP; and (c) PP tightened up the partnership within pairs. These findings suggest that teachers should take partnership into account as an important factor in PP or other collaborative learning, and adopt PP as an effective approach to decrease the gender gap in programming courses, and make students socialize.

1. Introduction

Programming for K-12 can be traced back to the 1960s when Logo programming was first introduced as an intellectual thinking educational tool for teaching mathematics (Feurzeig, Papert & Lawler, 2011). After Logo, the use of programming to teach thinking skills in K-12 was scarcely reported. In recent years, however, there has been renewed interest in introducing programming to K-12 students (Grover & Pea, 2013; Kafai & Burke, 2013). This was fuelled by the availability of easy-to-use visual programming languages such as Scratch (Brennan & Resnick, 2012; Burke, 2012; Lee, 2010), Stagecast Creator (Denner, Werner, & Ortiz, 2012) and Alice (Bishop-Clark, Courte, Evans, & Howard, 2006; Graczyńska, 2010; Kelleher & Pausch, 2007).

During programming, students are exposed to computational thinking (CT), a term popularized by Wing (2006). CT involves solving problems, designing systems, and understanding human behaviors, by drawing on the concepts fundamental to computer science (Wing, 2006). The nuts and bolts in CT are defining abstractions, working with multiple layers of abstraction and understanding the relationships among the different layers (Wing, 2008). Many researchers thought CT is a fundamental skill for almost everyone in a digital age, not just for computer scientists (National Research Council, 2011, 2010, pp. 3–4; Wing, 2006). More importantly, CT is in line with many aspects of 21st century competencies such as creativity, critical thinking, and problem solving (Binkley et al., 2012). Thus, it is not surprising that many educators claim that programming provides an important context and set of opportunities to develop CT for K-12 students (Brennan & Resnick, 2012; Kafai & Burke, 2013; Lye & Koh, 2014; Resnick et al., 2009).

This revived interest in programming for K-12 settings suggests a need to consider how CT can be fostered effectively via programming. Studies showed that students taught with pair programming (PP) performed better in CT than solo programming (Lye & Koh, 2014; Werner & Denning, 2009; Werner, Denner, Campe, & Kawamoto, 2012). The PP is a practice in which two people work side-by-side at one computer, and intensely collaborate to create a program. One is normally the “driver”, who is responsible for using a computer to key in codes. The other is usually known as the “navigator”, and takes the responsibility for observing the driver's work and providing support by pointing errors or offering ideas in solving a problem (Williams & Kessler, 2000).

In view of the usefulness to foster CT, we have used PP as a pedagogical teaching technique in a primary school for two years. Meanwhile, we identified some questions with putting PP in

* Corresponding author. School of Education Science, Nanjing Normal University, No. 122, Ninghai Road, Nanjing, 210097, China.
E-mail address: zhongbc@163.com (B. Zhong).

http://dx.doi.org/10.1016/j.chb.2016.07.017
0747-5632/

practice. One main question is how to pair up students to get effective teaching, which is about how the variation of pair formations may lead to different teaching results. There are several factors to consider for pairing up students, including students' skills, experiences, personalities, genders and partnership. In this study, we focused on the social factors including gender and partnership when pairing up students in the PP practice.

2. Literature review

Many studies have showed that PP has obvious benefits over solo programming, including but not limited to the aspects as follows. PP can (1) significantly improve individual programming skills and promote productivity or program quality since it provides students with a clearly defined method to help one another understand the context of the problem and to reflect on the contribution of both programmers (Braught, Eby, & Wahls, 2008; Cliburn, 2003; Hannay, Dybå, Arisholm, & Sjøberg, 2009; Li, Plaue & Kraemer, 2013; Tomayko, 2002; Williams & Kessler, 2000; Williams, Kessler, Cunningham, & Jeffries, 2000); (2) reduce frustration experienced by novice programmers, increase student satisfaction, enjoyment, and foster positive attitudes in programming (McDowell, Werner, Bullock, & Fernald, 2006, 2002; Bishop-Clark, Courte, Evans, & Howard, 2006; DeClue, 2003; LeJeune, 2006; Preston, 2005); (3) increase retention of students (especially for female students) in computer science courses (Li et al., 2013; McDowell et al., 2006); and (4) emphasize the importance of communication, teamwork, cooperation, and adaptability in a technical environment to better prepare students to work as a team (Cliburn, 2003; Williams & Kessler, 2000).

However, the above benefits do not occur automatically. Some experiments and empirical studies reported contradictory results to PP (Balijepally, Mahapatra, Nerur, & Price, 2009; Sfetsos, Stamelos, Angelis, & Deligiannis, 2009). There were many factors impacting the effectiveness of PP such as:

(a) Task complexity (Arisholm, Gallis, Dybă, & Sjoberg, 2007; Hannay, Arisholm, Engvik, & Sjøberg, 2010);
(b) Partners' skills and experiences (Hannay et al., 2010; Lui & Chan, 2006; Williams, Layman, Osborne, & Katira, 2006);
(c) Partners' learning styles (Salleh, Mendes, & Grundy, 2011; Williams et al., 2006; Zualkernan, Allert, & Qadah, 2006); and
(d) Partners' personalities and temperaments (Choi, Deek, & Im, 2008; Hannay et al., 2010; Katira et al., 2004; Sfetsos et al., 2009; Thomas, Ratcliffe, & Robertson, 2003; Williams et al., 2006).

The factors (b), (c) and (d) are directly related to the pair formation. For example, students seemed to do their best work when paired with students of similar levels of self-esteem (Thomas et al., 2003). We found, however, that most of these empirical studies were based only on competency or personality factors, largely ignoring the social factors (gender, partnership, race, and culture etc.) involved. This one-sided consideration possibly induced inadequacies in the pair formation selection, and illustrated an incomplete picture for PP.

In a family of newer studies, social factors especially gender aspect were included in the experiments' variables system. Choi (2015) claimed that one particular topic that has not received much attention is the gender topic which is about how men and women interact and collaborate in programming and dyad team work. Using a pool of university programming courses students as the experiment participants, the study examined three gender pair types: female–female, female–male, and male–male. The result revealed that there was no significant gender difference in the PP coding output. But the same gender pair exhibited significantly higher levels of pair compatibility than the mixed gender pair.

Katira, Williams, and Osborne (2005) conducted a study involving 361 software engineering students at North Carolina State University to understand and predict pair compatibility. The results indicated that pairing up a female student with a partner who has a similar SAT/GRE/GPA will likely result in a compatible pair. Minority students perceived compatibility with a partner who has a similar GPA. Pairs with different genders were less likely to report compatibility. When both students in the pair were minority students, they were more likely to perceive compatibility. Minority students were comfortable working with students of the same gender.

Lewis (2011) conducted a study to investigate differences between PP and collaborative learning in two summer enrichment classes for students entering the sixth grade. Although this study did not investigate the partnership, they thought the interruptions and opportunities for discussion reinforced the pair relationship because they observed that a student would infrequently consult another student who was not his/her partner, even if he/she was the same distance away as the student's partner.

McDowell et al. (2006) collected data from 554 students who attempted the programming course at the University of California-Santa Cruz. Each participant was paired with a preferred partner. The study reported (1) those who paired produced significantly better programs than those who worked alone, but there was no significant gender difference as a whole in average programming scores; (2) men were significantly more confident than women in solo programming, however, the 24% increase in confidence that pairing afforded women was even greater than the 15% confidence boost experienced by men who had the benefit of PP. The result was a significant decrease of a gender gap in confidence when working in PP.

In this section we summarized important findings from empirical studies on PP, especially from those studies that include psychology and gender factors in their research frameworks. There still exist many issues that have not been empirically investigated, such as the impact of student partnership on compatibility, confidence and performance in PP. Considering the fact that PP is one of the major human-centric software development paradigms, social factors, especially gender and partnership, need to be further addressed. Therefore, this paper reported an experimental investigation of two social factors (i.e. gender and partnership) and their interaction on PP with an expectation of making students have a better effectiveness toward programming. In addition, PP showed promise for reducing gender differences among college students, while this study was initiated to examine this promising practice in the primary school setting.

3. Research aim and questions

The study aimed to explore the impacts of two social factors on PP effectiveness. PP effectiveness is expressed in terms of PP result, measured in this study by compatibility of pairs, performance of student programming, confidence toward programming, and partnership of pairs. Therefore, four research questions were to be answered:

(a) Is there any difference on compatibility among the gender pairs and partnership pairs?
(b) Is there any difference on learning performance among the gender pairs and partnership pairs?
(c) Is there any difference on confidence toward programing among the gender pairs and partnership pairs? and