Australian Journal of Teacher Education

Volume 42 | Issue 3 Article 4

2017

Improving the Computational Thinking Pedagogical Capabilities of School Teachers

Matt Bower
Macquarie University, matt.bower@mq.edu.au

Leigh N. Wood
Macquarie University, leigh.wood@mq.edu.au

Jennifer W.M. Lai
Macquarie University, jennifer.lai@mq.edu.au

Cathie Howe
Macquarie ICT Innovations Centre, catherine.howe@det.nsw.edu.au

Raymond Lister
University of Technology Sydney, raymond.lister@uts.edu.au

See next page for additional authors

Recommended Citation
Bower, M., Wood, L. N., Lai, J. W., Howe, C., Lister, R., Mason, R., Highfield, K., & Veal, J. (2017). Improving the Computational Thinking Pedagogical Capabilities of School Teachers. *Australian Journal of Teacher Education, 42*(3).
http://dx.doi.org/10.14221/ajte.2017v42n3.4

This Journal Article is posted at Research Online.
http://ro.ecu.edu.au/ajte/vol42/iss3/4

Improving the Computational Thinking Pedagogical Capabilities of School Teachers

Authors
Matt Bower, Leigh N. Wood, Jennifer W.M. Lai, Cathie Howe, Raymond Lister, Raina Mason, Kate Highfield, and Jennifer Veal

This journal article is available in Australian Journal of Teacher Education: http://ro.ecu.edu.au/ajte/vol42/iss3/4

Improving the Computational Thinking Pedagogical Capabilities of School Teachers

Matt Bower
Leigh N. Wood
Jennifer W.M. Lai
Kate Highfield
Jennifer Veal
Macquarie University
Cathie Howe
Macquarie ICT Innovations Centre
Raymond Lister
University of Technology Sydney
Raina Mason
Southern Cross University

Abstract: The idea of computational thinking as skills and universal competence which every child should possess emerged last decade and has been gaining traction ever since. This raises a number of questions, including how to integrate computational thinking into the curriculum, whether teachers have computational thinking pedagogical capabilities to teach children, and the important professional development and training areas for teachers. The aim of this paper is to address the strategic issues by illustrating a series of computational thinking workshops for Foundation to Year 8 teachers held at an Australian university. Data indicated that teachers' computational thinking understanding, pedagogical capabilities, technological know-how and confidence can be improved in a relatively short period of time through targeted professional learning.

Introduction

This study investigated the challenge of developing teachers' computational thinking pedagogical capabilities, in order to determine how teachers can be best supported through professional development and resource provision. There has been a growing global interest in introducing computing into the school curriculum (Liu *et al.*, 2011). Several international professional bodies and initiatives have called for more attention to computational thinking in the curriculum (Voogt *et al.*, 2015). In particular, the National Science Foundation has assembled educational leaders to bring the concept of computational thinking to the K-12 classroom in the US (Barr, Harrison, & Conery, 2011). In Australia, it is mandated that all children from Foundation to Year 8 will learn computational thinking in their curriculum, according to the Australian Digital Technologies Syllabus (ACARA, 2012). In the UK, the change in the school curriculum for students from 5 – 16 also puts a strong focus on computational thinking (Sentance & Csizmadia, 2015).

In general, computational thinking has been defined as "*The process of recognising aspects of computation in the world that surrounds us, and applying tools and techniques from Computer Science to understand and reason about both natural and artificial systems*

and processes" (The Royal Society, 2012, p.29). It is seen as an important competency because today's students will not only work in fields influenced by computing, but also need to deal with computing in their everyday life and in today's global economy (Grover & Pea, 2013). At the same time, computer scientists discuss the need to increase students' interest in computer science through paying attention to computational thinking (and not only programming) in the compulsory curriculum in order to offer students' the option to continue their further studies or their career in fields related to computer science (Wolz *et al.*, 2011). Indeed, the most prevalently cited rationale in the literature for including computing in K-12 instruction is the growing demand for workers with computer science skills (Wilson & Moffat, 2010).

The process of increasing student exposure to computational thinking in schools is complex, requiring systemic change, teacher engagement and development of significant resources (Barr & Stephenson, 2011). With the new changes, teachers inevitably have to face challenges. If teachers have inaccurate and native perceptions of computational thinking, it will directly influence how they teach this area (Milton, Rohl, & House, 2007). Researchers have established strong connections between teacher efficacy and teacher behaviours that foster student achievement (Goddard, Hoy & Hoy, 2000; Guo *et al.*, 2010; Ross, Hogaboam-Gray, & Hannay, 2001). If teachers do not feel efficacious for teaching computational thinking, students may have negative experiences in learning the concept (Israel *et al.*, 2015). In addition, there is no widespread agreement about strategies for teaching and assessing the level of computational thinking development in students (Brennan & Resnick, 2012). Teachers may not be sure what kind of pedagogies are better for the purposes of teaching computational thinking.

In this paper, a comprehensive assessment of the difficulties and challenges in teaching computational thinking is performed so as to inform the design of computational thinking resources and professional development programs that best cater to the needs of teachers.

Computational Thinking: Literature Review

Researchers argue that there is no clear-cut definition of computational thinking (Hu, 2011). The idea of computational thinking was first introduced by Papert (1996), who is widely known for the development of the Logo software. In 2006, Wing defined computational thinking as "*Solving problems, designing systems, and understanding human behaviour, by drawing on the concepts fundamental to computer science*" (Wing, 2006, p. 33). Prompted by her article and a growing community of researchers, educators, and policymakers, computational thinking as a concept and associated research agenda has witnessed increasing attention and investigation (Grover & Pea, 2013). While computational thinking draws upon concepts that are fundamental to computing and computer science, it also includes practices such as problem representation, abstraction, decomposition, simulation, verification, and prediction (Sengupta *et al*., 2013). These practices, in turn, are also central to modelling, reasoning and problem-solving in a large number of scientific and mathematical disciplines (National Research Council, 2010). Einhorn (2012) states that computational thinking develops a variety of skills (logic, creativity, algorithmic thinking, modelling/simulations), involves the use of scientific methodologies and helps develop both inventiveness and innovative thinking. Academics in the field of education, in particular educational technology, agree with the computer science education community that computational thinking is an important, essential and very truly 21st-century skill (Einhorn, 2012; Voogt *et al.*, 2015; Wing, 2006).

Based on the definitions and core concepts of computational thinking as provided by computer scientists and researchers, several definitions have emerged for what computational thinking is in the domain of compulsory schooling globally. Key in all these definitions is the focus on students' complex problem-solving skills and dispositions (e.g. Barr & Stephenson, 2011; Lee *et al.,* 2011; Sengupta *et al.,* 2013) with the help of computing and computers (Grover & Pea, 2013; Wolz *et al.,* 2011). Mishra and Yadav (2013) have argued that computational thinking goes beyond typical human-computer interactions within the school curriculum; instead, they argued that human creativity could be augmented by computational thinking, in particular with the use of automation and algorithmic thinking (Van Dyne & Braun, 2014). However, researchers argue despite the best efforts to articulate that computational thinking is more than just "*programming*", the misconception that the two are the same remains (Lu & Fletcher, 2009; Qualls & Sherrell, 2010).

Based on their studies of novice interactive media designers, Brennan and Resnick (2012) have developed a framework for analysing computational thinking that is comprised of three key dimensions. The first dimension is "*computational concepts*", being the fundamental elements people use as they program, i.e., elements such as sequence, loops, parallelism, events, conditionals, operators and data. "*Computational practices*", the second dimension is defined as the processes people undertake as they engage with the concepts, such as abstracting and modularizing, reusing and remixing, testing and debugging, etc. The last dimension is "*computational perspectives*" and this dimension is related to the views people form about the world around them and about themselves with relation to computational thinking problem solving. Wood, Thomas and Rigby (2011) consider this sort of professional understanding (i.e., "*knowing for*") as the highest level of knowledge that is critical for successful participation in the workforce. Without an appreciation of all three dimensions it is difficult for teachers to help students understand the concepts, practices and applications of the discipline, which in turn may adversely affect their enjoyment and success in the domain.

Previous studies about computational thinking have largely focused upon students (e.g. Barr & Stephenson 2011; Grover & Pea, 2013; Yadav *et al.*, 2011). For instance, exploratory investigations by Lewandowski *et al* (2007) demonstrated how exposure to computational thinking enhances the way students approach problems. Several researchers propose that computational thinking can serve as effective vehicles for learning subjects like science and math concepts (Blikstein & Wilensky, 2009; Kynigos, 2007; Sengupta *et al.,* 2013). Werner, Denner and Campe (2015) tested new performance assessment from a student's perspective for measuring computational thinking in middle school.

Research on the integration of computational thinking in education is still scarce (Voogt *et al.,* 2015; Wilson & Guzdial, 2010). Many of the studies work on computational thinking focused mostly on definitional issues, and tools that foster computational thinking development (Grover & Pea, 2013). While computer science education researchers have recently contributed a significant amount of work to a growing knowledge base about teaching and learning computational thinking, studies do not often focus on teachers' perspective (Portelance & Bers, 2015). Similarly, there is little research that has systematically and comprehensively examined the influence of computational thinking on pre-service and in-service teachers (Yadav *et al.*, 2011).

Teaching Computational Thinking and the Challenges Involved

Multiple studies have shown that teachers have a profound effect on student learning (Whittle, Telford, & Benson, 2015) and different teaching strategies will affect the student achievement (Schroeder *et al.*, 2007). In general, educators have divided teaching strategies into two main types, i.e., teacher-centred and student-centred. In the teacher-centred classroom, teachers introduce the specific things that are worthy of being studied, and students are told how to interpret them. That is, students must learn—memorise—a meaning as dictated by the things that teachers introduce (Kauchak & Eggen, 1993). In the student-centred classroom, however, students are responsible for finding reasons that they can use to create knowledge and understanding (Pham, 2016). To teach computational thinking teachers requires a variety of different teaching approaches (Guzdial, 2008). At times teacher-centred approaches are beneficial to introduce concepts and model capabilities, however, it is critical that student-centred pedagogies are utilised in order for learners of computing to consolidate understanding, transfer their knowledge, develop their creativity, and have opportunities to learn from peers (Bower, 2011; Bower & Hedberg, 2010). It is suggested through continued professional development, teachers could be able to better adapt their computational thinking pedagogies and approaches based on student needs (Stevens *et al.*, 2013).

To truly integrate computational thinking into current primary and secondary curricula undoubtedly presents significant challenges, especially for teachers. Experts voiced concerns about a shortage of teachers qualified to deliver the new curriculum whenever new ideas and concepts are introduced (Peng *et al.*, 2014). Teachers express challenges that they face when teaching computational thinking concept and computing-related subjects (Grover & Pea, 2013). They are anxious when having to develop new learning resources and use novel technologies in class (Meerbaum-Salant, Armoni, & Ben-Ari, 2013). Some teachers are not confident in dealing with new and unfamiliar teaching materials (Curzon *et al.*, 2009). The lack of confidence is correlated with the low self-efficacy in teaching the subject (Sandholtz & Ringstaff, 2014). Self-efficacy is defined by Bandura as "*beliefs in one's capabilities to organize and execute the courses of action required to produce given attainments*" (1997, p.3). Low self-efficacy impacts on teaching performance and effectiveness (Babaei & Abednia, 2016). Previous studies have applied the self-efficacy concept to investigate teachers' pedagogical practices (Kreijns *et al.*, 2013; Paraskeva, Bouta, & Papagianni, 2008). It is clear that self-efficacy beliefs can positively influence teachers' pedagogical practices relating to the use of technology (Ertmer & Ottenbreit-Leftwich, 2010; Ertmer *et al.*, 2012). Results also indicate that teachers with high level of self-efficacy are more committed to teaching and have higher job satisfaction (Gunning & Mensah, 2011; Chen & Yeung, 2015). Thus, the benefits of developing teachers' self-efficacy with respect to computational thinking are potentially numerous.

Teachers also have issues relating to insufficient resources and support for teaching computational thinking skills (Sentance & Csizmadia, 2015). Providing teachers with adequate teaching-related resources is critical in order to improve student outcomes (Stevens *et al.*, 2013). Black *et al.* (2013) suggest that creating communities of practice could allow teachers to share the resources and provide each other with ongoing support.

Method

In order to develop teachers' computational thinking understanding, pedagogies, and dispositions, as well as to examine the malleability of each construct, a workshop was run in October of 2015. The workshop aimed to help Foundation to Year 8 teachers develop their

computational thinking pedagogies. The design and implementation of the workshops was funded through Google's Computer Science for High Schools (CS4HS) program. CS4HS is an annual grant program promoting computer science education worldwide by connecting educators to the skills and resources they need to teach computational thinking concepts in relevant ways (CS4HS, 2017). Workshops drew upon contemporary innovative practices from across the world as part of previous CS4HS programs, as well as prevailing literature in the computer science education field to progress the capabilities of teachers to teach the upcoming Australian Digital Technologies Curriculum. To measure teachers' understanding of and attitudes towards computational thinking, a two-stage survey, the pre-workshop and post-workshop teacher surveys were conducted. Three research questions guided this study:

1. What do teachers believe to be the different elements of computational thinking?
2. What support (and alternatively inhibits) the development of teachers' computational thinking pedagogical capabilities?
3. What conceptual and attitudinal change did teachers experience in terms of computational thinking pedagogies as a result of the workshops?

Workshops

The workshops were offered run at the Macquarie ICT Innovations Centre and were divided into four different full-day sessions for K-2, Year 3-4, Year 5-6, and Year 7-8 teachers. After defining computational thinking and relating it to the new Australian Digital Technologies Curriculum, participants were led through four modules relating to problem decomposition, patterns, abstraction, and algorithms. Each module started by explaining and modelling the key ideas behind the aspect of computational thinking being examined, before quickly moving onto activities that provided teachers with an experiential understanding of the phenomenon. Generally speaking the activities included an "*unplugged*" activity that used paper or other tactile modelling to demonstrate the area of computational thinking, followed by one or more technological activities. The technologies that were used varied according to the age and stage being targeted. For example, whereas the K-2 workshop used Beebots and Scratch Junior, the Stage 3 workshop used Kodu and Hopscotch.

Videos and examples were used to promote the relevance of computational thinking in society and for the workplace. Facilitators made efforts to establish a safe and supportive and collegial learning environment so that teachers felt comfortable sharing their emerging understanding of computational thinking concepts and pedagogies. Several opportunities were provided for discussion to encourage sharing of ideas and reflection.

Data Collection

The data for this study was collected in October of 2015. Two sources of data, pre-workshop and post-workshop teacher surveys were used to address the research questions. The pre-survey was issued to participants in the week before or on the morning of the session to determine their initial knowledge and dispositions. Apart from demographic questions relating to age, gender, and previous studies of computing, the survey asked about teachers' understanding of the term computational thinking, the pedagogical strategies they had for teaching computational thinking, and the technologies that could be used to develop computational thinking. Thus, this component of the survey was framed around their TPACK understanding (Mishra & Koehler, 2006). The survey instrument also inquired as to affective and systemic factors, namely, their confidence to teach computational thinking, the extent to

which they felt computational thinking was an important part of the curriculum, and the professional support they believed would assist their future endeavours. Confidence to teach computational thinking and the extent to which it was perceived to be important were measured using seven point Likert scales with response items ranging from "*strongly disagree*" to "*strongly agree*". The large majority of the participants completed the post-survey directly following the workshop. Participants who did not complete the survey immediately returned their responses within one week of the workshop. The post-survey included essentially identical questions to the pre-survey in order to gauge shifts in teacher understanding and perceptions that resulted from the workshops. It was deemed unlikely that any of the participants completed any other computational thinking professional learning or activities between surveys, allowing changes in knowledge and attitude to be attributed to the workshop they had completed.

Participants

A total of 91 and 75 teachers participated in the pre- and post-workshops surveys respectively. In this study only the results from respondents who participated both pre- and post-workshop surveys were included in the qualitative and quantitative analysis. There were 69 valid responses that contained complete responses that were used for the study. There were 50 females and 19 males in the samples with an average of 12.6 years of teaching experience. Among 69 teachers in the sample 44% indicated that they had completed some prior courses in computing, and this range from one day course in coding to a master degree in computing. The age range of the samples was from 18 to 64, with an average age of approximately 40 years old.

Data Analysis

The researchers conducted the data analysis in 2016. Open-ended questions of the pre- and post- workshop surveys were analysed using qualitative thematic analysis techniques in order to identify 1) concepts relating to computational thinking (content, pedagogy, and technology), 2) difficulties in learning new pedagogies, and 3) supports teachers needed. For each of these areas an open coding phase was used to determine preliminary analytic categories, followed by an axial-coding phase to determine emergent themes and a selective-coding phase to support reporting (as outlined by Neuman, 2006). All coding took place using the Computer Assisted Qualitative Data Analysis Software NVivo 11, which enabled aggregation of categories and themes across participants and cases, as well as the development of perceptions over time. A simple word frequency query was performed to get some ideas into the key themes emerging from the initial coding process. The rating was primarily undertaken by one member of the research team using a scheme that had been developed as part of parallel study into pre-service teacher conceptions of computational thinking (authors, in preparation). Consultation with other members of the research team was undertaken in cases where the correct coding category was ambiguous. After three rounds of comparing and line-by-line analysis all the themes for five open-ended questions were finalised. The responses were coded under two or more themes if the response incorporated multiple elements (Leech & Onwuegbuzie, 2011). The responses were ranked from the highest to the lowest number of occurrences under the respective categories.

The closed-ended questions (i.e., the quantitative data) of teacher pre- and post-workshop surveys were analysed by SPSS Statistics 22.0. Paired sample t-tests were used to determine whether statistically significant differences existed between the means of the pre-

and post-workshops about the teachers' perceptions on the importance of children develop computational thinking capabilities and the level of confidence about developing students' computational thinking capabilities. Results of the qualitative and quantitative analyses of the data are provided below.

Results

The results for this section are organised as follows. First, teacher awareness of the computational thinking and the Digital Technologies Syllabus is reported to provide an indication of the need for professional learning. Next, teacher understanding of the computational thinking concept, computational thinking pedagogies, and computational thinking technologies both before and after the workshop are presented, to illustrate how the workshop influenced their TPACK understanding (Mishra & Kohler, 2006). Finally, affective shifts in terms of teacher confidence and attitude towards computational thinking are reported.

Awareness of Computational Thinking and the New Curriculum

As aforementioned, computational thinking has been emphasised in the latest Australian Digital Technologies Syllabus. Table 1 shows the teachers' awareness of the term computational thinking and the upcoming Australian Digital Technologies Curriculum before the workshops. Among the 69 respondents, the results indicated that 26 of them were aware of the new Digital Technologies Curriculum and the term computational thinking (Table 1). There were 9 teachers who had not heard of the term computational thinking and were unaware of the new technologies curriculum. Thus in this study, it appeared necessary to increase some teachers' awareness of computational thinking and related concepts in order to help them better prepare for the new curriculum changes.

		Awareness of the term computational thinking		
		Yes	**No**	**Total (%)**
Aware of the new Australian Digital Technologies Curriculum	**Yes**	26 (38%)	28 (41%)	54 (78%)
	No	6 (9%)	9 (13%)	15 (22%)
	Total	**32 (46%)**	**37 (54%)**	**69 (100%)**

Table 1: Awareness of the term computational thinking and Australian Digital Technologies Curriculum before the workshops

Computational Thinking Concepts

Table 2 summaries the responses to the open-ended question "*What does computational thinking mean to you?*" before and after the workshops. Both before and after the workshop the 69 teachers generally regarded computational thinking as computational practices relating to "*problem-solving*" activities. One distinctive difference between pre- and post-workshops responses was that the detail of computational thinking definitions was more elaborate after the workshops (141 references pre-workshop versus 312 references post-workshops). This is evidence that respondents had a more evolved conception of computational thinking following the workshop. After the workshops many teachers could clearly identify computational thinking as comprising four key cornerstones (i.e.,

decomposition, pattern recognition, abstraction and algorithms). For instance, in the post-survey one respondent defined computational thinking as:

> *"A problem-solving process made up of four components, decomposition, patterns, abstraction algorithm."*

There was some evidence of computational thinking perspective development as a result of the workshop, as the following two examples indicate:

> *"Computational thinking means to use the computer sciences in order to solve problems or to create systems which improve outcomes in the real world - an important skill for students to acquire in order for them to have a skill set which will make them attractive to future employers."*
>
> *"After now having a greater understanding what computational thinking is then I am able to recognize its use in many aspects of life."*

However, generally speaking teachers did not indicate an awareness of the "*computational concepts and perspectives*" when explaining computational thinking, even after the workshops. This indicates that there may have been an opportunity to expand many teachers' conceptions and perspectives of computational thinking as part of the workshops to encompass actual constructs used during computing processes and relationships to the real world.

Pre-Workshop		**Post-Workshop**	
Computational thinking construct	**N**	**Computational thinking construct**	**N**
Computational practices	**101**	**Computational practices**	**279**
• Problem-solving	32	• Problem decomposition	62
• Logical thinking	16	• Problem-solving	60
• Writing scripts or coding	12	• Pattern recognition	46
• Algorithm development	7	• Abstraction	42
• Creative thinking	7	• Algorithm development	38
• Open-ended questioning	6	• Critical thinking	10
• Problem decomposition	6	• Logical thinking	9
• Organising data	4	• Writing scripts or coding	7
• Pattern matching	4	• Creative thinking	2
• Analytical thinking	3	• Decision making	2
• Designing systems	2	• Testing and debugging	1
• Abstraction	1	**Computational concepts**	**17**
• Critical thinking	1	• Sequences	14
Misconceptions	**24**	• Data	3
• Using technologies (generally)	13	**Computational perspectives**	**13**
• Programming	6	• Tackling real-world issues	7
• Thinking in the way a computer thinks	5	• Skills development	4
Computational concepts	**7**	• 21st Century Skills	2
• Sequences	4	**Others**	3
• Conditionals	1	No computer is needed	3
• Data	1		
• Events	1		
Computational perspectives	**6**		
• 21st Century skills	4		
• Tackling real-world issues	2		
Others	**3**		
• Unsure	2		
• Have not heard computational thinking before	1		
Total	**141**	**Total**	**312**

Table 2: Summary of computational thinking constructs before and after the workshops

Prior to the workshops there were some misconceptions about the meaning of computational thinking. Some teachers considered computational thinking as using technologies generally, computer programming, or thinking in a way a computer thinks:

"using technology to enhance learning"

"coding, multimedia, collaboration via web2"

"Learning how a computer thinks/does not think/Learning to program a computer"

Encouragingly, none of the participants had these or any other misconceptions of computational thinking after the workshops.

Pedagogical Strategies in Developing Computational Thinking Capabilities

In regard to the open-ended question "*What pedagogical strategies do you have (or can you think of) for developing school students' computational thinking capabilities?*", respondents mentioned a wide range of teaching strategies and those strategies could be categorised into two main areas, i.e., student-centred and teacher-centred pedagogies.

Rather than referring to the conventional instructional or teacher-centred approach, the most common pedagogical strategy both before and after the workshops was student-centred, the problem-solving approach (Table 3). However, the number of student-centred pedagogies mentioned by the respondents was more than double in the post-workshops survey, and the descriptions of the strategies were more detailed. It is argued that student-led educational experiences that are active and engaging lead to deeper learning (Adler, Whiting, & Wynn-Williams, 2004). In the post-workshop survey respondents also identified blended, flexible and challenge-based learning approaches.

Pre-Workshop		**Post-Workshop**	
Pedagogical strategies	**N**	**Pedagogical strategies**	**N**
Student-centred pedagogy	**59**	**Student-centred pedagogy**	**132**
• Problem-solving approach	25	• Problem-solving learning – using four cornerstones of computational thinking	88
• Student-oriented learning approach (e.g. peer-to-peer learning, support students' learning, skills development)	15	• Student-oriented learning (e.g. develop students' thinking abilities, self-reflection tasks, skills development)	18
• Open-ended tasks	8	• Open-ended tasks	6
• Project-based learning	5	• Project-based learning	5
• Group work and collaboration	3	• Group work and collaboration	5
• Inquiry-based learning	3	• Inquiry-based learning	4
Teacher-centred pedagogy	**30**	• Challenged-based learning approach	
• Basic usage of technologies and software applications	12	• Blended learning and flexible approach	
• Programming or coding related activities with instructions	10	**Teacher-centred pedagogy**	29
• Direct instruction, modelling	8	• Basic usage of technologies and software applications	16
Others	**10**	• Direct instruction, modelling	8
• Not sure	9	• Programming or coding related activities with instructions	5
• Embedded in the curriculum	1	**Others**	**31**
		• Embedded in all Key Learning Areas (KLAs) and curriculum	16
		• Any pedagogy can be used to teach CT	8
		• Others	4
		• Not sure	3
Total	**99**	**Total**	**192**

Table 3: Summary of computational thinking pedagogical strategies

The student-oriented learning approach emphasises student responsibility and activity in learning including developing students' thinking abilities, providing self-reflection tasks and skills development, rather than focusing upon what the teachers are doing. Teachers also referenced using group work to teach computational thinking. Furthermore, the respondents believed that computational thinking could be embedded in all key learning areas (KLAs) and some of them proposed that "*any pedagogy can be used to teach computational thinking*".

Supportive Technologies

Regarding the question "*What technologies can be used to develop school students' computational thinking capabilities?*", "*using computer and digital devices*" was the most common response. There were 41 and 50 responses from the pre- and post-workshops respectively agreeing technological devices could support the development of computational thinking. Devices like personal computers, iPads, mobile phones, laptops, interactive whiteboards, interactive televisions, digital cameras, etc. were referenced by teachers, particularly in the pre-survey responses. Researchers comment that digital devices, systems and networks foster computational approaches to solving problems (Grover & Pea, 2013). Many teachers, especially after attending the workshops replied that they would use computer programming and coding technologies to develop computational thinking knowledge. Whereas only 42% of the teachers were able to identify specific software before the workshops, after the workshops 72% of the teachers were able to name specific software like Scratch Jnr, Visual Basic, Python, Hopscotch, Tynker, etc. which students could use for coding and programming. Teachers also suggested using robotics to develop computational thinking capabilities. For instance, Beebots, and Lego Robotics were commonly identified by the teachers as useful tools for developing computational thinking skills.

Over 30 post-workshop responses suggested that games and apps could help students learn relevant skills. Games like Kodu, Minecraft and apps like iPad apps were frequently mentioned by the teachers. Teachers also indicated that they could also get the useful technologies online through Code.org. After the workshops, none of the respondents were unsure about the technologies that could be used to develop computational thinking.

Lack of Confidence

In regard to the question "*What prevents you from feeling confident about developing your students' computational thinking capabilities?*", teachers before the workshops indicated that "*lack of knowledge and inability to understand computational thinking*" was the main reason preventing them from feeling confident to help students' learning computational thinking (Table 4). For instance, teachers said that:

"Not knowing enough about what it is and what it involves EXACTLY."

"I don't really understand the definition of 'computational thinking' which is why I am attending the course."

Reasons for the lack of confidence were classified into two main categories. One was related to the insufficient resources and the other was more about the low self-efficacy of the teachers. Prior to the workshops, teachers indicated that they were unaware of the support, funding, activities and programs that were available (9 responses) to facilitate the teaching of computational concepts. They were concerned about the lack of time (7) and lack of support from the upper level and peers (3). However, most significant reason for the lack of confidence was due to the self-efficacy of the teachers (78). Most of the teachers complained

about the lack of knowledge and ability to understand the concepts of computational thinking (36) and lack of effective teaching strategies and experience (14).

The responses were, however, different after the workshops. Feeling incompetent to teach the concepts had become a less frequently reported issue, and lack of resources to support teaching had become more of an issue. When compared to the pre-workshop survey, there were more responses relating to the category of lack of resources (54), whereas responses for low self-efficacy were reduced from 78 to 45. Examples of concerns relating to lack of resources included:

> *"Resources available and the pedagogical skills to link it back to the curriculum and specific examples of where and how to use it."*
> *"Physical resources within school and duration of access to available resources."*
> *"Actually, the attitude of the profession (both in pre-service and out there in the field). There seems to be a lot of negativity on computational thinking and coding from the profession. Some warranted (like lack of resources and support for teachers) and some not (like saying it's all too hard or just another fad)."*

Several teachers indicated that the pacing of the curriculum was too fast to add computational thinking into hence they wanted to know how to integrate the concept into the curriculum.

> *"I have a very crowded curriculum and only see students 40-60 minutes a week, I would like to transfer this to other teachers to make it more integrated."*

Pre-Workshop		**Post-Workshop**	
Lack of confidence	**N**	**Lack of confidence**	**N**
Low self-efficacy	**78**	**Lack of resources**	**54**
• Lack of knowledge and ability to understand computational thinking	36	• Not aware of the support, resources, activities, programs that are available	17
• Lack of effective teaching strategies	14	• Lack of time	16
• Lack of experience, practice and training	13	• Lack of support from peers or colleagues	14
• How to integrate concepts into the Curriculum	12	• Lack of availability of technologies and infrastructures	7
• Can't keep up with every changing technologies	3	**Low self-efficacy**	**45**
	19	• Lack of knowledge and ability to understand computational thinking	18
Lack of resources		• Lack of experience, practice and training	13
• Not aware of the support, funding, activities, programs that are available	9	• Lack of effective teaching strategies	7
• Lack of time	7	• How to integrate concepts into the Curriculum	6
• Lack of support from peers and colleagues	3	• Can't keep up with every changing technologies and software environment	1
Others	**3**	**Others**	**2**
• Unsure	3	• Not much	2
Total	**100**	**Total**	**101**

Table 4: Summary of the reasons preventing teachers from feeling confident

The final qualitative question was "*What could help you to feel more confident about developing your students' computational thinking capabilities*?" Teachers after the workshops were more likely to think of ways to build their confidence and were clearer about what would be more helpful. In particular, teachers wanted to get more resources and support like providing more time to learn (15), accessing relevant technologies (e.g. iPads) (13); provision of more examples, activities and ideas on how to teach computational thinking (12)

and soliciting advice from colleagues (9) (Table 5). Teachers also felt that their confidence could be improved through networking with professionals or mentors (4). Example teacher quotations include:

> *"Having the time to explore different ways to introduce and use these in the classroom.*
> *Having my own iPad to spend time investigating apps and setting up lessons."*
> *"Projects that are more directly aligned with our curriculum. I think would give more teacher confidence to have a go."*

In addition, teachers requested formal professional development and training (38) in order to improve their teaching capabilities.

Pre-workshop		**Post-workshop**	
Building confidence	**N**	**Building confidence**	**N**
Resources and advice	**61**	**Resources and advice**	**74**
• Advice on effective teaching strategies	36	• Provide more time to learn	15
• What resources are available to students and teachers	9	• Get more relevant technologies and resources (e.g. hardware like iPads)	13
• Advice on how to incorporate the concepts into the Curriculum	9	• Provide examples, activities or lesson ideas	12
• Advice on how to keep up with every changing technologies	4	• Advice from peers, teaching buddy	9
• Learn from peers	3	• Advice on effective teaching strategies	8
Formal professional development and training	**24**	• Advice on how to incorporate the concepts into the Curriculum	6
• Attend courses and training	16	• Resources to understand the mechanics of programming apps and software	6
• Professional development	8	• Professional networks or mentoring	4
Others	**4**	• A team of teachers driving computational thinking	1
• Try new things, take risks	3	**Formal professional development and training**	**38**
• Ways to get other staff on board	1	• Attend courses and training	23
		• Professional learning	15
		Others	**2**
		• Learn with students	2
Total	**89**	**Total**	**114**

Table 5: Summary of the ways to help teachers feel more confident

Changes in Teachers' Attitude and Confidence

Two paired sample t-test were performed in order to test the teachers attitudes towards the importance of children develop computational thinking capabilities and the confidence of the teachers about developing their students' computational thinking capabilities before and after the workshops. Participants who previously completed the pre-workshop survey were included in the analyses.

In relation to the attitudes towards the importance of children develop computational thinking capabilities, the analysis revealed a significant difference between the mean scores of the participants before attending the workshops (M=6.32, SD=1.13) and after attending the workshops (M=6.76, SD=0.47); $t(61) = -3.32$, p=0.002 (Table 6).

Another paired sample t-test analysis indicated that the teachers felt more confidence to develop their students' computational thinking abilities after the workshops. There was a significant difference between the mean scores before the workshop (M=3.80, SD=1.22) and after the workshops (M=4.74, SD=0.68); $t(53) = -5.55$, p=0.000.

The results indicated that it is possible for teachers to build up their confidence level by attending workshops and enhance their computational thinking pedagogical capabilities in quite a short period of time.

Measures	Pre-Workshop Mean	Pro-Workshop Mean	*T*	*p*
It is important that children develop computational thinking capabilities.[#]	6.32	6.76	-3.32	0.002
How confident do you feel to develop your students' computational thinking capabilities? [##]	3.80	4.74	-5.55	0.000

[#] The measure was scored using a 7-point Likert scale (1 = Strongly Disagree; 7= Strongly Agree).
[##] The measure was scored using a 6-point Likert scale (1 = Extremely Unconfident; 6 = Extremely Confident).
All comparisons were two-tailed, paired-sample t-tests.

Table 6: Perception of the development of children computational thinking capabilities and confidence about developing students' computational thinking capabilities

Discussion

Preparing teachers to teach computational thinking and computing generally is a major challenge of our time (Liu *et al.*, 2011; Voogt *et al.*, 2015). It is an unfamiliar discipline to many teachers, and one that they have never previously had any formal instruction in, let alone instruction on how to teach it. This lack of knowledge is highlighted amongst the teachers in our sample, most of whom were not aware of the term computational thinking even though it is the foundational concept in the new Australian Digital Technologies Curriculum. This lack of knowledge has been documented elsewhere (Voogt *et al.*, 2015).

However, findings from this study indicate that teacher's computational thinking capabilities are relatively malleable. Analysing pre- and post- workshop definitions of computational thinking revealed that in general teachers had developed more specific and detailed understanding of computational thinking, including a better awareness of sub-components of computational thinking (for instance, decomposition, pattern recognition, algorithm design and abstraction). By the end of the one-day professional learning program the teachers also had much more specific and divergent ideas about the sorts of pedagogies that they could use to develop computational thinking, most of which focused on student-centred learning approaches. Whereas before the workshop teachers were generally unaware of domain specific technologies that could be used to teach computational thinking but after the session the large majority of teachers were able to identify specific software (e.g. Scratch Junior, Python, Tynker, Beebots, Kodu, etc.). This is encouraging in so far as it demonstrates that targeted short-term professional development can have an impact upon teacher computational thinking pedagogical capabilities. However, there was potentially an opportunity for teachers to develop a better understanding of computational thinking perspectives, as this level of knowledge is important in order to help students appreciate the real-life value of what they are learning (Wood *et al.*, 2011).

Coming into the workshops the majority (78%) of reasons for low self-confidence about teaching computational thinking related to self-efficacy – a lack of understanding of computational thinking and associated pedagogical know-how. Lack of resources was a less common issue (19%). By the end of the one-day professional learning program the reasons for teachers' low self-confidence had shifted more towards being about resources (53%), with less frequent concern about self-efficacy (45%). Whereas before the workshops over half (59%) of teachers felt they needed advice on effective teaching strategies, after the workshop

this had fallen to small minority of teachers (11%). Resources, including having time to learn and appropriate technological resources became much greater concerns. Following the workshop teachers also had an increased desire for further professional development and training. By the end of the professional development program the teachers has significantly improved their confidence to teach computational thinking ($p<0.001$). Based upon previous research on teacher self-efficacy and learning outcomes (Guo *et al.,* 2010, Ross *et al.,* 2001) there is a likelihood that these improvements in self-efficacy would translate into higher quality teaching in the classroom.

It was encouraging to see the increasing emphasis that teachers placed on student-centred pedagogies as a result of the workshop, however, teachers indicated several ways in which they require support going forward. These include advice on how to best incorporate computational thinking concepts into the curriculum, time and resources to build understanding of relevant technologies, and more time to learn and plan their lessons. Teachers also identified the utility of further professional development and establishing professional learning networks or mentoring relationships. These findings concur with previous work on preparing educators to teach computational thinking (Grover & Pea, 2013).

Other types of support can be given to the teachers with the purpose of enhancing their computational thinking pedagogical capabilities, apart from provision of professional learning and resources. At the local level, schools could organise support groups that include peer learning and teaching buddies for struggling teachers. In addition, schools could allocate additional resources (e.g. computers, time) for teachers to develop teaching materials that engage students and facilitate deep learning. Management should aim to be more supportive and initiate changes (Barajas, Kikis, & Scheueremann, 2003). The school curriculum is a complex environment where multiple competing priorities exist (Settle *et al.,* 2012), and thus it is difficult for the in-service teachers to change and add one big concept into the already busy curriculum without support from management. Academics and industry could do more to engage with teachers, especially if they wish to introduce deep computational thinking principles in schools (Black *et al.,* 2013).

This study also has implications for teaching and teacher education outside the discipline of computing. Even though the minority (44%) of the teachers in the study had not previously undertaken any study with relation to the discipline, the cohort as a whole was able to demonstrate objective increases in technological, pedagogical and conceptual (TPACK) knowledge, and significantly improve their teaching confidence. One criticism commonly wagered against teachers is that they do not have the capacity to adequately respond to new disciplinary and teaching demands that result from changing technological, workforce and social needs. But the results of this study indicate that teachers can in fact be quite responsive, and in fact the main issue may be providing them with well-designed professional learning opportunities and resources. If the self-efficacy of teachers is so critical to improving their classroom practice and student learning outcomes (Babaei & Abednia, 2016; Ertmer & Ottenbreit-Leftwich, 2010; Ertmer *et al.,* 2012), then principals and systems are well advised to prioritise extensive professional support that builds self-efficacy and capabilities.

This study has a few limitations that should be noted. One was the small sample size, and future research could be on a larger scale. The extent to which findings are generalisable is left to the discretion of the readers. We note that this sample was not selected using any particular criteria, but did self-select to undertake the workshop. In addition, this study took place in an Australian environment so findings from this study might not generalise to other school contexts, although Australian educational culture is similar in many ways to most western countries. International study is recommended to investigate teachers' difficulties in teaching computational thinking given that there has been a growing global interest in

introducing computing into teaching plans. Furthermore, future research could examine changes to teacher computational thinking pedagogical capability and self-efficacy over time. In other words, another wave of longitudinal studies performed 6 months after the workshops could help understanding which support should be provided to teachers for continuing development. Research could also investigate the transferability of the capabilities teachers develop into professional settings and classroom practice. It would also be beneficial to develop systematic and rigorous ways of examining and measuring programming and computational thinking performance in the following studies.

Conclusions

Computational thinking draws on skills and professional practices that are fundamental to computing and computer science (Sengupta *et al.*, 2013). It includes problem representation, abstraction, decomposition, simulation, verification, and prediction. Recently, arguments have been made in favour of integrating computational thinking and programming into the school curricula. This is especially pertinent if the goal is to reach a larger number of students for the anticipated information technology workforce shortage and also enable people to utilise computational thinking to solve problems in any field (Barr & Stephenson, 2011). Previous studies on the topic of computational thinking were mostly about the definitional issues and from students' perspectives (Portelance & Bers, 2015). Previous research has expressed the need for greater research on how to support integration of computational thinking into the curriculum (Portelance & Bers, 2015; Voogt *et al.*, 2015; Wilson & Guzdial, 2010).

This study filled a gap in the research by examining the malleability of in-service teacher computational thinking pedagogies and related issues. The results of the post-workshop survey showed that teachers were able to improve the basic ideas of computational thinking content, pedagogy and technology in a relatively short period of time, as well develop significantly better self-efficacy towards the related concepts and practices. With higher self-efficacy, teachers are more prepared and have more confidence to teach in general. In this study, teachers' computational thinking pedagogical capabilities could be enhanced in a relatively short period of time. This is encouraging given the enormity of preparing a teaching workforce to successfully teach a set of thinking skills and concepts that are unfamiliar to many of them. However, to better prepare them for the future they indicated resources, time, peer mentoring networks and additional targeted professional learning workshops will all be beneficial.

References

ACARA. (2012). *The shape of the Australian curriculum: Technologies*. http://www.acara.edu.au/verve/_resources/Shape_of_the_Australian_Curriculum_-_Technologies_-_August_2012.pdf

Adler, R. W., Whiting, R. H., & Wynn-Williams, K. (2004). Student-led and teacher-led case presentations: Empirical evidence about learning styles in an accounting course. *Accounting Education*, *13*(2), 213-229. https://doi.org/10.1080/0963928041000167662O

Babaei, M., & Abednia, A. (2016). Reflective teaching and self-efficacy beliefs: Exploring relationships in the context of teaching EFL in Iran. *Australian Journal of Teacher Education*, *41*(9). http://dx. https://doi.org/10.14221/ajte.2016v41n9.1

Bandura, A. (1997). Self-efficacy: The exercise of control. New York: W.H. Freeman.

Barajas, M., Kikis, K., & Scheueremann, F. (2003). Monitoring and evaluation of research of learning innovations with ICT: Qualitative indicators of change. In M. Barajas (Ed.), *Learning innovations with ICT: Socio-economic perspectives in Europe* (pp. 15-46). Barcelona: McGraw-Hill Interamericana de Espana, S.A.U.

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology*, 38(6), 20-23.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads*, 2(1), 48-54. https://doi.org/10.1145/1929887.1929905

Black, J., Brodie, J., Curzon, P., Myketiak, C., McOwan, P. W., & Meagher, L. R. (2013, July). Making computing interesting to school students: teachers' perspectives. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education* (pp. 255-260). ACM. https://doi.org/10.1145/1929887.1929905

Blikstein, P., & Wilensky, U. (2009). An atom is known by the company it keeps: A constructionist learning environment for materials science using agent-based modeling. *International Journal of Computers for Mathematical Learning*, 14(2), 81-119. https://doi.org/10.1007/s10758-009-9148-8

Bower, M. (2011). Redesigning a web-conferencing environment to scaffold computing students' creative design processes. *Educational Technology & Society*, *14*(1), 27-42.

Bower, M., & Hedberg, J. G. (2010). A quantitative multimodal discourse analysis of teaching and learning in a web-conferencing environment–the efficacy of student-centred learning designs. *Computers & Education*, *54*(2), 462-478. https://doi.org/10.1016/j.compedu.2009.08.030

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada* (pp. 1-25).

Chen, Z., & Yeung, A. S. (2015). Self-efficacy in teaching Chinese as a foreign language in Australian schools. *Australian Journal of Teacher Education*, 40(8). http://dx. https://doi.org/10.14221/ajte.2015v40n8.2

CS4HS. (2017). *Developing CS educators globally for today's 21st century students.* http://cs4hs.com/index.html

Curzon, P., McOwan, P. W., Cutts, Q. I., & Bell, T. (2009, July). Enthusing & inspiring with reusable kinaesthetic activities. In *ACM SIGCSE Bulletin* (Vol. 41, No. 3, pp. 94-98). ACM. https://doi.org/10.1145/1595496.1562911

Einhorn, S. (2012). Microworlds, computational thinking, and 21st century learning. *LCSI White Paper*. http://www.microworlds.com

Ertmer, P. A., & Ottenbreit-Leftwich, A. T. (2010). Teacher technology change: How knowledge, confidence, beliefs, and culture intersect. *Journal of Research on Technology in Education*, *42*(3), 255-284. https://doi.org/10.1080/15391523.2010.10782551

Ertmer, P. A., Ottenbreit-Leftwich, A. T., Sadik, O., Sendurur, E., & Sendurur, P. (2012). Teacher beliefs and technology integration practices: A critical relationship. *Computers & Education*, *59*(2), 423-435. https://doi.org/10.1016/j.compedu.2012.02.001

Goddard, R. D., Hoy, W. K., & Hoy, A. W. (2000). Collective teacher efficacy: Its meaning, measure, and impact on student achievement. *American Educational Research Journal*, 37(2), 479-507. https://doi.org/10.3102/00028312037002479

Grover, S., & Pea, R. (2013). Computational thinking in K–12. A review of the state of the field. *Educational Researcher*, 42(1), 38-43. https://doi.org/10.3102/0013189X12463051

Gunning, A. M., & Mensah, F. M. (2011). Preservice elementary teachers' development of self-efficacy and confidence to teach science: A case study. *Journal of Science Teacher Education*, *22*(2), 171-185. https://doi.org/10.1007/s10972-010-9198-8

Guo, Y., Piasta, S. B., Justice, L. M., & Kaderavek, J. N. (2010). Relations among preschool teachers' self-efficacy, classroom quality, and children's language and literacy gains. *Teaching and Teacher Education*, *26*(4), 1094-1103. https://doi.org/10.1016/j.tate.2009.11.005

Guzdial, M. (2008). Education paving the way for computational thinking. *Communications of the ACM*, 51(8), 25-27. https://doi.org/10.1145/1378704.1378713

Hu, C. (2011, June). Computational thinking: What it might mean and what we might do about it. In *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education* (pp. 223-227). ACM. https://doi.org/10.1145/1999747.1999811

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, 82: 263-279. https://doi.org/10.1016/j.compedu.2014.11.022

Kauchak, D. P., & Eggen, P. D. (1993). *Learning and teaching*. Boston: Needham.

Kreijns, K., Van Acker, F., Vermeulen, M., & Van Buuren, H. (2013). What stimulates teachers to integrate ICT in their pedagogical practices? The use of digital learning materials in education. *Computers in Human Behaviour*, 29(1), 217-225. https://doi.org/10.1016/j.chb.2012.08.008

Kynigos, C. (2007). Using half-baked microworlds to challenge teacher educators' knowing. *International Journal of Computers for Mathematical Learning*, 12(2), 87-111. https://doi.org/10.1007/s10758-007-9114-2

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, 2(1), 32-37. https://doi.org/10.1145/1929887.1929902

Leech, N. L., & Onwuegbuzie, A. J. (2011). Beyond constant comparison qualitative data analysis: Using NVivo. *School Psychology Quarterly*, *26*(1), 70. https://doi.org/10.1145/1929887.1929902

Lewandowski, G., Bouvier, D. J., McCartney, R., Sanders, K., & Simon, B. (2007, September). Commonsense computing (episode 3): Concurrency and concert tickets. In *Proceedings of the Third International Workshop on Computing Education Research* (pp. 133-144). ACM. https://doi.org/10.1145/1288580.1288598

Liu, J., Lin, C. H., Hasson, E. P., & Barnett, Z. D. (2011, March). Introducing computer science to K-12 through a summer computing workshop for teachers. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (pp. 389-394). ACM.

Lu, J., & Fletcher, G. H. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*, *41*(1), 260-264. https://doi.org/10.1145/1539024.1508959

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education*, 23(3), 239-264. https://doi.org/10.1080/08993408.2013.832022

Milton, M., Rohl, M., & House, H. (2007). Secondary beginning teacher's preparedness to teach literacy and numeracy: A survey. *Australian Journal of Teacher Education*, *32*(2). https://doi.org/10.14221/ajte.2007v32n2.4

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record*, *108*(6), 1017-1054. https://doi.org/10.1111/j.1467-9620.2006.00684.x

Mishra, P., & Yadav, A. (2013). Of art and algorithms: Rethinking technology & creativity in the 21st century. *TechTrends*, 57(3), 11.

National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking*. Washington, DC: National Academies Press.

Neuman, W. L. (2006). *Social research methods: Quantitative and qualitative approaches*. MA: Allyn and bacon Boston.

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, 1(1), 95-123. https://doi.org/10.1007/BF00191473

Paraskeva, F., Bouta, H., & Papagianni, A. (2008). Individual characteristics and computer self-efficacy in secondary education teachers to integrate technology in educational practice. *Computers & Education*, 50(3), 1084-1091. https://doi.org/10.1016/j.compedu.2006.10.006

Peng, W. J., McNess, E., Thomas, S., Wu, X. R., Zhang, C., Li, J. Z., & Tian, H. S. (2014). Emerging perceptions of teacher quality and teacher development in China. *International Journal of Educational Development*, 34, 77-89. https://doi.org/10.1016/j.ijedudev.2013.04.005

Pham, T. (2016). Student-centredness: Exploring the culturally appropriate pedagogical space in Vietnamese higher education classrooms using activity theory. *Australian Journal of Teacher Education*, 41(1). https://doi.org/10.14221/ajte.2016v41n1.1

Portelance, D. J., & Bers, M. U. (2015, June). Code and tell: assessing young children's learning of computational thinking using peer video interviews with ScratchJr. In *Proceedings of the 14th International Conference on Interaction Design and Children* (pp. 271-274). ACM. https://doi.org/10.1145/2771839.2771894

Qualls, J. A., & Sherrell, L. B. (2010). Why computational thinking should be integrated into the curriculum. *Journal of Computing Sciences in Colleges*, *25*(5), 66-71.

Ross, J. A., Hogaboam-Gray, A., & Hannay, L. (2001). Effects of teacher efficacy on computer skills and computer cognitions of Canadian students in grades K-3. *The Elementary School Journal*, 102(2), 141-156. https://doi.org/10.1086/499697

Sandholtz, J. H., & Ringstaff, C. (2014). Inspiring instructional change in elementary school science: The relationship between enhanced self-efficacy and teacher practices. *Journal of Science Teacher Education*, *25*(6), 729-751. https://doi.org/10.1007/s10972-014-9393-0

Schroeder, C. M., Scott, T. P., Tolson, H., Huang, T. Y., & Lee, Y. H. (2007). A meta-analysis of national research: Effects of teaching strategies on student achievement in science in the United States. *Journal of Research in Science Teaching*, 44(10), 1436-1460. https://doi.org/10.1002/tea.20212

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18(2), 351-380. https://doi.org/10.1007/s10639-012-9240-x

Sentance, S., & Csizmadia, A. (2015). Teachers' perspectives on successful strategies for teaching Computing in school. *Paper presented at IFIP TCS*, 2015. http://community.computingatschool.org.uk/files/6769/original.pdf

Settle, A., Franke, B., Hansen, R., Spaltro, F., Jurisson, C., Rennert-May, C., & Wildeman, B. (2012, July). Infusing computational thinking into the middle-and high-school curriculum. In *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education* (pp. 22-27). ACM. https://doi.org/10.1145/2325296.2325306

Stevens, T., Aguirre-Munoz, Z., Harris, G., Higgins, R., & Liu, X. (2013). Middle level mathematics teachers' self-efficacy growth through professional development: Differences based on mathematical background. *Australian Journal of Teacher Education*, *38*(4). https://doi.org/10.14221/ajte.2013v38n4.3

The Royal Society (2012). *Shut down or restart? The way forward for computing in UK schools*. https://royalsociety.org/~/media/Royal_Society_Content/education/policy/computing-in-schools/2012-01-12-Computing-in-Schools.pdf

Van Dyne, M., & Braun, J. (2014, March). Effectiveness of a computational thinking (cs0) course on student analytical skills. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (pp. 133-138). ACM. https://doi.org/10.1145/2538862.2538956

Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 20(4), 715-728. https://doi.org/10.1007/s10639-015-9412-6

Werner, L., Denner, J., & Campe, S. (2015). Children programming games: a strategy for measuring computational learning. *ACM Transactions on Computing Education (TOCE)*, 14(4), 24.

Whittle, R. J., Telford, A., & Benson, A. C. (2015). The 'perfect' senior (VCE) secondary physical education teacher: Student perceptions of teacher-related factors that influence academic performance. *Australian Journal of Teacher Education*, 40(8). https://doi.org/10.14221/ajte.2015v40n8.1

Wilson, A., & Moffat, D. C. (2010, September). Evaluating Scratch to introduce younger schoolchildren to programming. In *Proceedings of the Psychology of Programming Interest Group Workshop, Madrid/Espanha*.

Wilson, C., & Guzdial, M. (2010). How to make progress in computing education. *Communications of the ACM*, *53*(5), 35-37. https://doi.org/10.1145/1735223.1735235

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35. https://doi.org/10.1145/1118178.1118215

Wolz, U., Stone, M., Pearson, K., Pulimood, S. M., & Switzer, M. (2011). Computational thinking and expository writing in the middle school. *ACM Transactions on Computing Education (TOCE)*, 11(2), 9. https://doi.org/10.1145/1993069.1993073

Wood, L. N., Thomas, T., & Rigby, B. (2011). Assessment and standards for graduate outcomes. *Asian Social Science*, 7(4), 12. https://doi.org/10.5539/ass.v7n4p12

Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011, March). Introducing computational thinking in education courses. In *Proceedings of the 42nd ACM technical symposium on Computer science education (pp. 465-470). ACM*. https://doi.org/10.1145/1953163.1953297

Acknowledgements

We would like to thank Google for generously financing the computational thinking workshops through their fantastic Computer Science for High Schools (CS4HS) scheme. We would also like to acknowledge the Macquarie ICT Innovations Centre for their support in running the workshops.