Chapter 8
Computational Thinking in K-12: In-service Teacher Perceptions of Computational Thinking

Phil Sands, Aman Yadav, and Jon Good

8.1 Introduction

Much of what we know about computational thinking comes from early research in educational practices using computers (Papert 1980; Pea and Kurland 1984) and from common conceptions of how computer scientists think about problems designed to be solved by computers (Denning 2009). Wing (2006) formalized computational thinking in an influential article discussing the ways computer scientists think about problems and how skills associated with computing are broadly applicable in other disciplines. Wing sparked a discussion about how educators should prepare students for careers influenced by computing and where core computational thinking concepts could be integrated into K-12 curricula (Barr and Stephenson 2011; Grover and Pea 2013; Yadav et al. 2014). Almost a decade later, teaching computational thinking skills to students has permeated at all levels of elementary and secondary schools. This integration is being done through the generation of new curricula within computer science education programs – the AP computer science principles course is one notable example – as well as in other content areas, such as mathematics and science (Weintrop et al. 2016). With this increased interest, however, comes key questions about how in-service teachers conceptualize computational thinking, especially teachers who are not trained in computer science. Namely, how do these teachers understand computational concepts as they work to apply them in their classrooms? Further, what steps do we need to take to help in-service teachers integrate computational thinking into their curriculum?

Most of the attention on embedding computational thinking during the past decade has focused on preservice teachers (Yadav et al. 2011, 2014). While this

P. Sands · A. Yadav (✉) · J. Good
College of Education, Michigan State University, East Lansing, MI, USA
e-mail: ayadav@msu.edu

M. S. Khine (ed.), *Computational Thinking in the STEM Disciplines*,
https://doi.org/10.1007/978-3-319-93566-9_8

information can help guide in-service teachers' professional development, we have yet to identify the unique challenges that exist in introducing computational thinking to non-computing teachers. A better understanding of in-service teachers' conceptions of computational thinking can guide design of teacher professional development programs. In a recent survey, we examined how K-12 in-service teachers perceive computational thinking within elementary and secondary classrooms. We present results from the survey and provide recommendations for developing professional development programs around computational thinking practices. We also discuss specific areas within the computational thinking model that lend themselves to the nature of applied problem-solving in K-12 classrooms.

8.2 Background

In considering computational thinking and its application to student preparation, Wing (2008) pointed to the links between CT and the wide variety of disciplinary skills traditionally taught in K-12 classrooms. These connections focus on the ubiquitous nature of computing and the nature of abstraction as it pertains to STEM career pathways. In addition, Wing stressed that computational thinking was not the same as the practice of programming; rather, she argued that the skills used in programming are useful for problem-solving in multiple contexts. Denning (2009) argued for the use of computational thinking ideas as the "third leg of science," a component of the inquiry process as much as it is a separate and distinct discipline. While Wing and Denning differed in how computational thinking was framed, they both agreed on the benefits for students from learning computer science. Regardless of which perspective one takes, it is apparent that the connections between computing and K-12 curricula are deep enough to justify the interest in further embedding these ideas in classrooms.

Since Wing (2006) introduced computational thinking, there have been several attempts to expand on what ideas encapsulate CT. Wing proposed that computational skills include abstraction, problem decomposition, pattern recognition, algorithmic thinking, and logical thinking. In attempting to draw connections between these skills and an educational model in Bloom's taxonomy, Selby (2015) organized a variation of these ideas by perceived difficulty: evaluation, algorithm design, generalization, abstraction of functionality, abstraction of data, and decomposition. Barr and Stephenson (2011) proposed nine major computational thinking concepts and abilities to be used within K-12 classrooms across core content areas. These include data collection, data analysis, data representation, problem decomposition, abstraction, algorithms and procedures, automation, parallelization, and simulation. This set is echoed in the work of Grover and Pea (2013), who offered that CT was comprised of abstractions and pattern generalizations, systematic processing of information, symbol systems and representations, algorithmic notions of flow of control, structured problem decomposition, iterative, recursive, and parallel

thinking, conditional logic, efficiency and performance constraints, and debugging and systematic error detection. A more complex set of skills were described by the National Research Council (2010) including:

> reformulation of difficult problems by reduction and transformation; approximate solutions; parallel processing; checking and model checking as generalizations of dimensional analysis; problem abstraction and decomposition; problem representation; modularization; error prevention, testing, debugging, recovery and correction; damage containment; simulation; heuristic reasoning; planning, learning, and scheduling in the presence of uncertainty; search strategies; analysis of the computational complexity of algorithms and processes; and balancing computational costs against other design criteria. (p. 3)

Given the wide variety of skills that can be connected to computational thinking, the lack of a clearly defined subset of skills may confuse educators trying to implement these practices.

Computational thinking skills have also appeared in recent updates to K-12 curriculum frameworks, such as Next Generation Science Standards (NGSS) as well as other curricula designed to teach introductory computing skills. The Next Generation Science Standards (NGSS) include the use of CT as an important practice to develop scientific understanding (NGSS Lead States 2013). The College Board created a new Advanced Placement computing course focusing on six key computational thinking practices, with the goal of attracting a more diverse group of students to computer science (2014). Similarly, Google introduced the CS First initiative to provide traditional computer science activities and lessons focused on computational thinking primarily for use by out-of-school organizations.

Considering that the onus for implementing these programs is on educators with limited experience in computing, a concern is the risk of conflating computational thinking with computer science or mathematics. There is also a potential for those implementing computational thinking ideas to imply that both CT and CS require the use of programming in all contexts (Fletcher and Lu 2009). In order to address this issue, it has been suggested that educators encourage the use of computational thinking skills at an early age, concentrating more on the innate thought processes that are associated with computing as opposed to specific computing tools. By doing so, educators can reduce the barriers for entry for students taking computing courses later in their academic careers (Margolis et al. 2010). This group includes not just students that develop further interest in computer science but also students interested in other fields engaging with computing in some form.

In spite of the potentially overwhelming set of skills that can be included in definitions of computational thinking, it is possible to implement most of the core ideas in primary and secondary classrooms without overemphasizing technical abilities. Examples can include digital storytelling, simple data collection, and the encouragement of scientific investigation (Lee et al. 2014). Considering that teachers may be using these skills in primary school classrooms already (Mannila et al. 2014), this suggests a need to help move teachers from implicit to explicit practices grounded in an understanding of why computational practices are relevant to student development.

8.3 Need

Computational thinking practices have the potential to develop student interest in how computing plays a role in other disciplines, specifically STEM. In order to see the benefits of student exposure to these computing concepts, we need to train both preservice and in-service teachers in computational thinking practices regardless of academic discipline. Across the United States, academic standards have been rewritten to include computational thinking as a core principle of curriculum implementation. Examples of this include the Next Generation Science Standards which include computational thinking concepts (NGSS 2013), Indiana's K-8 science standards (Indiana Department of Education 2017), and Texas' Essential Knowledge and Skills for elementary education (Texas State Board of Education 2012). Designing teacher professional development program should focus on augmenting teachers existing competencies while relying on established best practices, in order to align courses with the major components of computational thinking. As an important step in this process, we need to understand in-service teachers' current perceptions of computational thinking (Prieto-Rodriguez and Berretta 2014). In identifying areas of need, the transition can then be made to connecting professional development with classroom integration of CT. This study examined in-service teachers' conceptions of computational thinking and was guided by the following research questions:

1. How do in-service teachers conceptualize computational thinking as it would manifest in classroom practice?
2. How does teachers' subject area influence their computational thinking conceptualizations?
3. How does teachers' grade level taught influence their computational thinking conceptualizations?

8.4 Methods

Participants Seventy-four elementary and secondary teachers from a Midwestern state participated in the study. Of these teachers, 65 were female and 9 were male. Teachers taught at a variety of levels in the K-12 spectrum but could be divided roughly into primary school (N = 45) and secondary school (N = 29) levels. For the purposes of this study, we included grades K-6 as primary school teachers and grades 7–12 as secondary school teachers. Lastly, we considered those teachers that taught primarily STEM subjects (N = 29) versus those that were in non-STEM subjects (N = 55). STEM subjects included mathematics, science, computers, or technology.

Survey The survey included ten Likert scale questions based on prior work examining preservice teachers' perceptions of computational thinking (Yadav et al. 2011, 2014). The survey items began with the phrase "Computational thinking involves..." followed by a short stem that either belonged or did not belong to the broader perception of computational thinking. Teachers responded to the items on a Likert scale with five potential response values. These included "strongly agree," "agree," "disagree," "strongly disagree," and "don't know." Table 8.1a includes the list of survey items, and Table 8.1b includes how we characterized whether the item aligned with literature's conceptions of computational thinking. It should be noted in this table that the concept of "coding/programming" was not categorized due to disagreement over whether programming is an essential element of teaching CT in classrooms (Denning 2009; Wing 2006; Brennan and Resnick 2012). The internal reliability of these items was assessed using Cronbach's alpha ($\alpha = 0.92$). In addition, the survey included items to collect demographic information regarding teachers' gender, grade level taught, and subjects taught.

The survey was distributed at the Michigan Association for Computer Users in Learning (MACUL) conference. Participants were recruited at an exhibition booth for university K-12 outreach programming.

Table 8.1a Items included in the teacher survey

Computational thinking involves...
... solving problems
... using heuristics/algorithms
... logical thinking
... thinking like a computer
... coding/programming
... doing mathematics
... using computers (e.g., office tools)
... knowing how to use a computer
... using technology in your teaching
... playing online games

Table 8.1b How researchers categorized items from the teacher survey

Computational thinking involves...	Computational thinking does not involve...
... solving problems	... doing mathematics
... using heuristics/algorithms	... using computers (e.g., office tools)
... logical thinking	... knowing how to use a computer
... thinking like a computer	... using technology in your teaching
	... playing online games
It is unclear whether or not computational thinking involves...	
... coding/programming	

8.5 Data Analysis

Likert response was given a numerical value from 1 to 4 ("strongly agree," 1; "agree," 2; "disagree," 3; "strongly disagree," 4), and missing responses and those marked as "don't know" were excluded from these calculations. We used descriptive analysis for each of the survey items to view patterns in teachers' conceptions of computational thinking. In addition, Mann-Whitney U test was used to analyze the influence of teachers' subject area and grade level taught on their conceptions of computational thinking. Mann-Whitney U test, a nonparametric alternative test to the independent t-test, was used due to the ordinal nature of the data. The data was analyzed using the R statistical package.

8.6 Results

Majority of the teachers in our study were most confident that computational thinking involved logical thinking (100%), doing mathematics (100%), and solving problems (99%). To a lesser degree, majority of the teachers also agreed that computational thinking involved using heuristics or algorithms (93%), using computers (86%), using technology in teaching (82%), and knowing how to use a computer (76%). Teachers' conceptions of computational thinking are shown in Fig. 8.1, and the descriptive statistics are presented in Table 8.2.

8.6.1 STEM vs Non-STEM Teachers

STEM refers to teaching and learning in the fields of science, mathematics, engineering, and technology (Gonzalez and Kuenzi 2012). For the purpose of this study, teachers that specified their primary area as one of the natural sciences or engineering (e.g., computer science, physics, chemistry, etc.) were included within STEM. This group was categorized as "STEM" teachers, and those outside of these disciplines was categorized as "non-STEM" teachers. For this study, most of the primary school teachers were removed from the STEM analysis because these educators commonly teach all domains. Only those primary educators that specified a domain specialization were considered in this analysis. Table 8.3 shows the breakdown by grade level and STEM specialization.

As shown in Fig. 8.2, results showed that STEM teachers had the greatest confidence that computational thinking involved doing mathematics (100%), logical thinking (100%), solving problems (100%), using computers (96%), and using heuristics or algorithms (96%). The non-STEM teachers showed similar beliefs that computational thinking involved doing mathematics (100%), logical thinking (100%), solving problems (100%), and using heuristics or algorithms (93%). While

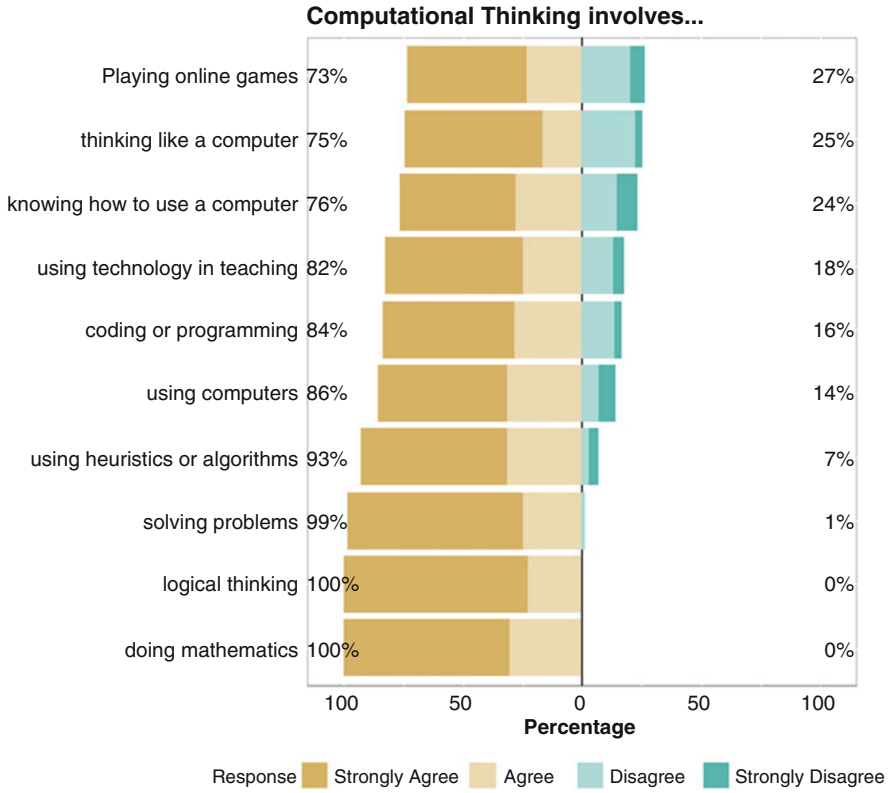

Fig. 8.1 Teachers' conceptions of computational thinking

Table 8.2 Descriptive statistics on teachers' conceptions of computational thinking

Computational thinking involves...	Mean	Standard deviation
... doing mathematics	1.31	0.46
... using computers (e.g., office tools)	1.67	0.90
... solving problems	1.28	0.48
... using heuristics/algorithms	1.5	0.76
... logical thinking	1.23	0.42
... thinking like a computer	1.70	0.92
... knowing how to use a computer	1.84	0.99
... using technology in your teaching	1.65	0.88
... playing online games	1.83	0.97
... coding/programming	1.64	0.83

Note: The scale was from 1 (strongly agree) to 4 (strongly disagrees)

Table 8.3 Primary and secondary teachers considering STEM vs non-STEM teaching credentials

	Primary	Secondary	
STEM	14	15	29
Non-STEM	31	14	45
	45	29	

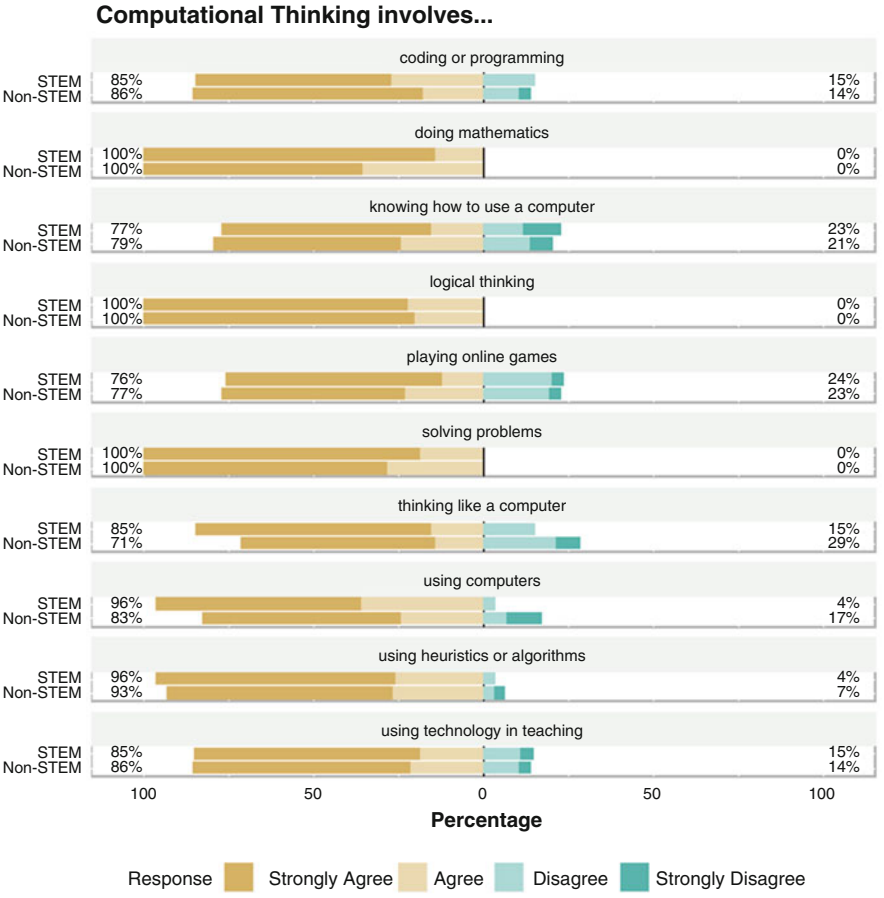

Fig. 8.2 STEM vs. non-STEM teachers and perceptions of computational thinking

there were similar responses between the STEM and non-STEM teachers on almost all of the items, two notable exceptions were "thinking like a computer" and "using computers." This showed that non-STEM teachers were less likely to view those as computational thinking. It should be noted that "using computers" was described on the survey instrument as being akin to using office tools and other applications.

Table 8.4 Mann-Whitney U test comparing STEM vs Non-STEM teachers

Computational thinking involves...	U statistic	p-value
... doing mathematics	526	0.06557
... using computers	437.5	0.5706
... solving problems	473.5	0.3973
... using heuristics or algorithms	423.5	0.7236
... logical thinking	396	0.8475
... thinking like a computer	420	0.2644
... knowing how to use a computer	389	0.8276
... using technology in teaching	385	0.8967
... playing online games	349	0.6169
... coding or programming	333	0.5387

Mann-Whitney U results exhibited there was no significant difference between STEM and non-STEM teachers on how they conceptualized computational thinking (see Table 8.4 for the Mann-Whitney U statistics for each of the computational thinking items).

8.6.2 Primary vs Secondary School Teachers

Over the last decade, the high awareness of STEM curricula has led to more elementary teachers exploring ways to engage their students in technology (DeJarnette 2012); hence, we examined whether there were differences in how they conceptualized computational thinking when compared to secondary teachers. As shown in Fig. 8.3, results demonstrated that secondary teachers believed that computational thinking involved doing mathematics (100%), logical thinking (100%), solving problems (100%), and using heuristics or algorithms (100%). Similarly, primary teachers also viewed computational thinking as involving doing mathematics (100%), logical thinking (100%), and solving problems (98%). However, there were some differences between the two groups as secondary teachers disagreed at a higher rate whether computational thinking involved "knowing how to use a computer," "playing online games," and "using technology in teaching." In addition, they had uniform sentiment that "using heuristics or algorithms" belonged to computational thinking, while primary teachers showed some disagreement. Other items showed some differences, but none that were visually significant enough to note.

Mann-Whitney U results suggested no significant difference between primary and secondary teachers on how they conceptualized computational thinking (see Table 8.5 for the Mann-Whitney U statistics for each of the computational thinking items).

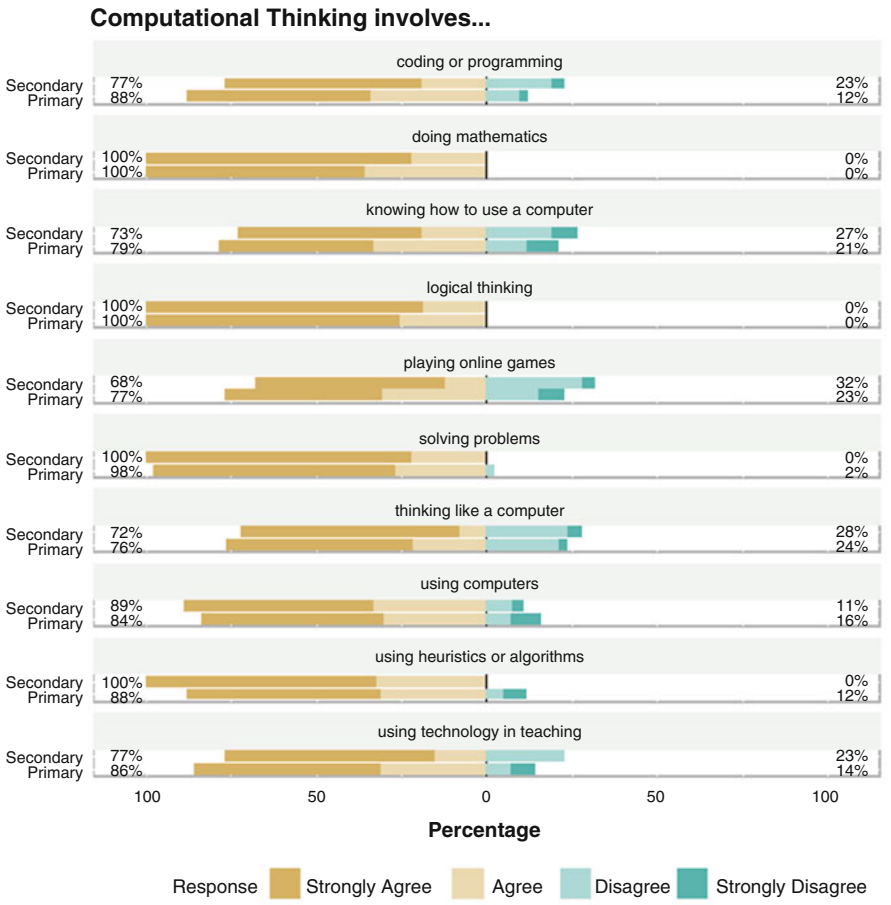

Fig. 8.3 Primary vs. secondary teachers and perceptions of computational thinking

Table 8.5 Mann-Whitney U test comparing primary and secondary teachers' perceptions

Computational thinking involves...	U statistic	p-value
... doing mathematics	688.5	0.24
... using computers	607	0.72
... solving problems	651	0.51
... using heuristics or algorithms	673.5	0.24
... logical thinking	621.5	0.50
... thinking like a computer	550.5	0.71
... knowing how to use a computer	571.5	0.73
... using technology in teaching	565	0.79
... playing online games	508.5	0.76
... coding or programming	525.5	0.92

8.7 Discussion

Overall, results suggested that while teachers conceptualized computational thinking in alignment with the literature, they also had some incorrect ideas about what computational thinking entailed. We also found that there were no differences on teachers' conceptions of computational thinking based upon either the content area (STEM vs. non-STEM) or grade level (primary vs. secondary). Computational thinking involves a set of skills that describe many of the same abilities inherent to programming and problem-solving with computers (Denning 2009). The responses given by the teachers in our study suggested that many educators have very little knowledge about what these skills are and lack awareness of how these skills can be implemented in their classrooms. The results suggest that there is much work to be done before in-service teachers are able to implement computational thinking in their classrooms.

Based on the literature, we classified what computational thinking entails (see Table 8.1b). Our results exhibited that teachers had the greatest confidence that CT involved "logical thinking" and "solving problems," which align with how computational thinking has been conceptualized recently (Denning 2017). On the other hand, teachers also viewed CT as "doing mathematics," which does not align with the common conception of computational thinking. Overall, we found that majority of the teachers strongly agreed with all the components of computational thinking outlined in the survey items and in many cases that teachers incorrectly agreed with concepts that we did not view as computational thinking. With these conceptions of computational thinking, a teacher simply using digital tools, such as Microsoft Office, might think that he/she is engaging his/her students in computational thinking. On the other hand, it is also possible that teachers might think that CT involves too many conceptual tasks to integrate.

Our results support the need to develop non-computing teachers' understanding of computational thinking if it is to permeate within K-12. Teachers, regardless of whether they taught a STEM subject or not, have similar ideas about computational thinking and sometimes hold incorrect conceptions. Given the prevalence of incorrect views related to computational thinking suggests that while CT maybe a buzzword in computing education, many teachers are not being introduced to the core components of computational thinking. While researchers have argued for the need to embed computational thinking within teacher education (Yadav et al. 2017), our results suggest the need to also train in-service teachers. This training needs to be content-specific on how to integrate computational thinking ideas into existing curriculum. Specifically, teachers need to be introduced to computational thinking in a way that meets their existing learning goals and fits within their pedagogical practices. Rather than adapting approaches designed for preservice teachers, we instead propose implementing a distinct strategy for integrating CT ideas aimed at teachers already working in K-12 classrooms.

In-service teacher professional programs need to provide support for content integration, allowing educators to utilize their existing body of knowledge while

also meeting their needs with regard to time constraints and availability. Existing research into teacher professional development has found the difficulties of providing long-term gains in the classroom based on limited exposure to applied concepts through isolated workshop sessions (Harris and Sass 2011; Desimone 2009). Thus, in order to successfully train teachers to integrate computational thinking into K-12 classrooms, we need to develop ongoing and continuous professional development programs that help teachers develop a thorough understanding about what it means to think computationally and then engage their students in computing ideas (Yadav et al. 2017).

Professional development needs to draw upon teachers' expertise in their content knowledge, pedagogical knowledge, and pedagogical content knowledge. The Reading Apprenticeship model (Greenleaf et al. 2011) provides a framework to support teachers' learning of computational thinking concepts and develop students' understanding of how computation can be applied in specific subject areas. Specifically, professional development should point out clear connections and how computational thinking can meet subject area learning goals rather than just being an instructional add-on in the K-12 curriculum (Greenleaf et al.). Given the large number of demands teachers face and the time constraints of the classroom, we also need to address how to deliver the content to teachers. Schools of education should collaborate with departments of computer science to lead state-approved professional development certification programs in computing education. These low-cost flexible programs could be delivered online, to allow teachers to learn virtually and be a member of an online community of practice to discuss how computational thinking can be embedded to meet their subject-specific learning goals. As suggested by Yadav et al. (2017), we believe that an online community of practice would allow teachers to effectively integrate computational thinking to meet their curriculum needs.

Our findings have important implications for how professional development programs should be structured to ensure that teachers effectively integrate computational thinking in their classrooms. Results suggest that professional development needs to differentiate between the use of computing tools and the concepts and practices inherent to computational thinking. It might be beneficial to expose teachers to computational thinking without the use of computers, such as using the CS Unplugged curriculum (Bell et al. 2009). Focusing on unplugged activities might help teachers grasp how computational thinking and the use of computers in the classroom differ from one another. We believe that given Wing's (2006) description of computational thinking overlapped with aspects of problem-solving components, such as abstraction, problem decomposition, pattern recognition, and algorithmic thinking, a focus on problem-solving skills offers a low floor to get teachers interested in computational thinking. By using problem-solving as the focus, we feel that more teachers will be motivated to embed subcomponents of computational thinking in their regular academic subjects (Yadav et al. 2016).

This study had a few limitations, which has implications for generalizability of the findings. First, we acknowledge that the survey was based on a small number of teachers and may not have accurately represented teacher knowledge of

computational thinking across the United States. The impact of this small group is also enhanced due to the large number of elementary teachers in our sample that were not included in our evaluation of STEM and non-STEM teachers. Additionally, given that participants in our study were volunteers might lead to self-selection bias, which limits generalizability of the results. It is also possible that the since teachers completed the survey at a conference focused on technology in education, they were more focused on computational thinking as involving use of technology/digital tools. At the same time, given that teachers interested in technology struggled with identifying computational thinking ideas suggests we have an uphill climb before CT becomes another core subject similar to reading, writing, and arithmetic as called for by Wing (2006).

In summary, we recognize the need to prepare students for twenty-first-century careers makes it essential for K-12 teachers to be prepared to integrate computational thinking concepts. This requires a multipronged approach to prepare teachers at the preservice and in-service level to become computationally literate.

References

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.

Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology, 13*(1), 20–29.

Brennan, K., & Resnick, M. (2012). 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).

The College Board. (2014). AP Computer Science Principles: 2016–2017. Retrieved from https://advancesinap.collegeboard.org/stem/computer-science-principles/course-details

DeJarnette, N. (2012). America's children: Providing early exposure to STEM (science, technology, engineering and math) initiatives. *Education, 133*(1), 77–84.

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM, 60*(6), 33–39. https://doi.org/10.1145/2998438.

Denning, P. J. (2009). Beyond computational thinking. *Communications of the ACM, 52*(6), 28–30.

Desimone, L. M. (2009). Improving impact studies of teachers' professional development: Toward better conceptualizations and measures. *Educational Researcher, 38*(3), 181–199.

Fletcher, G. H., & Lu, J. J. (2009). Education: Human computing skills: Rethinking the K-12 experience. Association for computing machinery. *Communications of the ACM, 52*(2), 23.

Gonzalez, H. B., & Kuenzi, J. J. (2012). *Science, technology, engineering, and mathematics (STEM) education: A primer*. Congressional research service. Retrieved from https://fas.org/sgp/crs/misc/R42642.pdf

Greenleaf, C. L., Litman, C., Hanson, T. L., Rosen, R., Boscardin, C. K., Herman, J., Schneider, S. A., Madden, S., & Jones, B. (2011). Integrating literacy and science in biology: Teaching and learning impacts of reading apprenticeship professional development. *American Educational Research Journal, 48*(3), 647–717.

Grover, S., & Pea, R. (2013). Computational thinking in K–12 a review of the state of the field. *Educational Researcher, 42*(1), 38–43.

Harris, D. N., & Sass, T. R. (2011). Teacher training, teacher quality and student achievement. *Journal of Public Economics, 95*(7), 798–812.

Indiana Department of Education (2017). *Indiana academic standards: Science & computer science*. Retrieved from http://www.doe.in.gov/standards/science-computer-science.

Lee, I., Martin, F., & Apone, K. (2014). Integrating computational thinking across the K–8 curriculum. *ACM Inroads, 5*(4), 64–71.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014). Computational thinking in K-9 education. In *Proceedings of the working group reports of the 2014 on innovation & technology in computer science education conference* (pp. 1–29). New York: ACM.

Margolis, J., Estrella, R., Goode, J., Holme, J. J., & Nao, K. (2010). *Stuck in the shallow end: Education, race, and computing*. Cambridge: MIT Press.

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

NGSS Lead States (Ed.). (2013). *Next generation science standards: for states, by states*. Washington, DC: National Academies Press. Retrieved from http://ezproxy.msu.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=e000xna&AN=867791.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books, Inc.

Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology, 2*(2), 137–168.

Prieto-Rodriguez, E., & Berretta, R. (2014). Digital technology teachers' perceptions of computer science: It is not all about programming. In *2014 I.E. Frontiers in Education Conference (FIE) Proceedings* (pp. 1–5).

Selby, C. C. (2015). Relationships: Computational thinking, pedagogy of programming, and bloom's taxonomy. In *Proceedings of the workshop in primary and secondary computing education* (pp. 80–87). New York: ACM.

Texas State Board of Education (2012). *Chapter 111. Texas essential knowledge and skills for mathematics subchapter a. Elementary.* Retrieved from http://ritter.tea.state.tx.us/rules/tac/chapter111/ch111a.html

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717–3725.

Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2017). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education, 26*, 235–254. https://doi.org/10.1080/08993408.2016.1257418.

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding a 21st century problem solving in K-12 classrooms. *TechTrends, 60*, 565–568. https://doi.org/10.1007/s11528-016-0087-7.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE), 14*(1), 5.

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). New York: ACM.