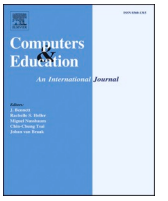
Computers & Education 186 (2022) 104530

Contents lists available at ScienceDirect

Computers & Education

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



Investigating the impact of research-based professional development on teacher learning and classroom practice: Findings from computer science education

Chrystalla Mouza [a,*], Diane Codding [b], Lori Pollock [c]

[a] School of Education, University of Delaware, Newark, DE, 19716, USA
[b] Department of Physics & Astronomy, Northwestern University, Evanston, IL, 60208, USA
[c] Computer & Information Sciences, University of Delaware, Newark, DE, 19716, USA

ARTICLE INFO

Keywords:
Computer science education
Professional development
Teacher learning

ABSTRACT

As the field of computer science (CS) is gaining increased attention, the need for qualified teachers is rapidly growing. Yet little is still known about the design features, implementation, and outcomes of professional development programs in computing. The purpose of this study is threefold: (a) examine a CS professional development program built around high-quality design features reported in the research literature, (b) investigate the impact of the program on participating teachers' learning and classroom practice, and (c) identify specific design features that facilitated changes in teacher learning and practice. The study employed a mixed-methods design. Data were collected from multiple sources including, pre and post survey data on teacher knowledge of CS content, pedagogy, and technology (N = 94), as well as interviews and classroom implementation data from eight case study participants. Findings from this work indicated that participants reported improvements in their knowledge of CS content, pedagogy, and technology. They also applied new learning into their practice, though implementation varied among participants. Responding on the value of the professional development design features, teachers noted the importance of focusing on CS content knowledge as well as opportunities to engage with pedagogical practices for teaching computing. Findings also indicated the important role of contextualized follow-up classroom support in the implementation of new learning into practice. These findings have implications for the design of professional development programs grounded in best practices with the potential to support broad efforts intended to prepare teachers with the knowledge and skills needed to deliver CS education.

1. Introduction

In recent years, there has been a growing interest in computer science (CS) education across all levels of the K-12 system. While schools in the U.S. have offered computing classes since the 1980s, the focus has been on *using* computers and software applications, rather than *designing* or *creating* computing innovations (Code.org et al., 2021). Recognizing the importance of helping students move from users to creators of computing, several countries around the world (e.g., Australia, England) have already adopted CS as a required school subject (Brown et al., 2014; Falkner et al., 2014). Similarly, policy initiatives in the U.S., such as *CS for All*, emphasize

* Corresponding author.
E-mail addresses: cmouza@udel.edu (C. Mouza), dcodding@northwestern.edu (D. Codding), pollock@udel.edu (L. Pollock).

https://doi.org/10.1016/j.compedu.2022.104530
Received 31 December 2021; Received in revised form 16 April 2022; Accepted 18 April 2022
Available online 22 April 2022

the importance of helping all students acquire knowledge and skills related to big ideas in CS. To support these efforts, new standards from the Computer Science Teachers Association (2017) and the International Society for Technology in Education (2016) encourage a shift toward the development of fundamental CS knowledge and skills among K-12 students. Additionally, knowledge and concepts from CS (e.g., computational thinking) are included in content specific standards such as the Next Generation Science Standards (NGSS, 2013), thus demonstrating the importance of CS across disciplines.

A key challenge in realizing the success of these efforts is teacher preparation and support. Despite growing interest in CS education, few teachers have the knowledge to deliver CS curricula or embed CS content into existing school curricula (Rich et al., 2021; Sadik et al., 2020). Effective CS teaching necessitates building teachers' knowledge of new content (CK), knowledge of good pedagogical practices (PK), and knowledge of technology (TK) for CS instruction (Vivian & Falkner, 2019). Interactions among these knowledge domains form the core of what Mishra and Koehler (2006) called Technological Pedagogical Content Knowledge (TPACK). Towards this end, professional development (PD) is essential for helping teachers navigate the knowledge domains needed to realize the promise of CS (Codding et al., 2021; Mouza et al, 2018; Vivian & Falkner, 2019). In this work, we examine a PD program built around high-quality design features aimed at helping teachers in grades 5–12 deliver CS instruction. The PD centers on two components: a week-long summer institute focusing on building teachers' knowledge, and follow-up classroom support provided by university undergraduates with a background in CS. Relatedly, we address three research questions:

1. How did participation in the summer institute influence teacher learning of CS content, pedagogy, and technology?
2. How did teachers who received follow-up classroom support implement new learning into CS practice?
3. How did specific PD design features influence teacher learning and implementation of CS content, pedagogy, and technology in their classrooms?

2. Literature review

2.1. Characteristics of effective PD

Effective PD encompasses five key elements, including focus on content, active learning, coherence, collaboration, and sustained duration (Darling-Hammond et al., 2017; Desimone, 2009). Specifically, PD needs to provide teachers with opportunities to improve their knowledge of content and pedagogical approaches. This element is critical in CS where teachers are new to the content and need opportunities to deepen their knowledge (Basu et al., 2021). To foster teacher learning, PD should include active learning strategies that provide teachers opportunities to observe, analyze student work, make presentations, and engage in continuous reflection (Darling-Hammond et al., 2017; Desimone & Garet, 2015). Importantly, PD should be coherent by helping teachers form connections with the materials they are expected to teach based on the specific needs of their students (Desimone & Garet, 2015). Specific to CS, it is important to help teachers who do not teach standalone CS courses form connections between CS and core curricular content (Basu et al., 2021).

In terms of structure, quality PD needs to support collaboration, be of sufficient duration, and provide follow-up support (Darling-Hammond et al., 2017). Collaboration is particularly important for CS teachers who often have no colleagues to ask for support (Goode et al., 2020). In these instances, allowing opportunities to build learning communities with teachers from other schools or districts is essential. Finally, effective PD extends over multiple days offering at least 20 contact hours and opportunities for follow-up support (Desimone & Garet, 2015). Follow-up classroom support is critical for overcoming isolation and supporting ongoing learning especially in a field like CS which advances rapidly (Margolis et al., 2017).

While there is wide recognition that the above elements are essential in teacher PD, much of the evidence comes from other domains, making it difficult to draw conclusions about what works for specific areas such as CS (Penuel et al., 2007). Additionally, most efforts that examine teacher learning outcomes from PD focus on data obtained through surveys that gauge teachers' opinions and attitudes related to the experience, rather than changes in their learning or pedagogy (Lawless & Pellegrino, 2007). In fact, in a review of the literature related to PD efforts in CS, Menekse (2015) found that only 10% of the reviewed studies utilized interviews in addition to surveys to address issues of implementation. To address this gap, we focus on three levels of evaluation (Guskey, 2013), including participants' reactions, learning, and application of PD learning in practice. We also utilize multiple measures including pre and post surveys on both learning and perceptions of PD, follow-up interviews, and classroom implementation data.

2.2. PD focusing on CS

Teacher PD is a critical step in efforts to provide engaging CS instruction (Yadav et al., 2016). Yet compared to other subjects, there are fewer opportunities and less research documenting the design, implementation, and outcomes of CS-related PD (Menekse, 2015; Vegas et al., 2021). In an extended literature review, Menekse (2015) uncovered just 21 studies of which only four were published in peer reviewed journals. Menekse (2015) found that most of the PD was offered by higher education institutions but lacked collaboration with K-12 districts. In terms of content, PD typically focused on integrating CS into other subject areas, teaching CS knowledge to teachers with little or no experience, and addressing issues of equity and broadening participation in computing. Importantly, Menekse's review found that most CS PD efforts were not consistent with high-quality PD principles reported in the literature (see Desimone, 2009). Relatedly, Basu et al. (2021), reported that CS PD is most often provided in a workshop or summer institute format with no follow-up support. Further, PD focuses primarily on deepening teachers' content knowledge with less attention on pedagogy (Basu et al., 2021).

One of the most widely known efforts to prepare teachers for CS is associated with the Exploring Computer Science (ECS) curriculum (Margolis et al., 2017). The ECS PD program centers on three key areas: inquiry-based learning, equity, and content. The program is delivered over a 2-year period, beginning with a one-week summer institute and continuing with follow-up Saturday sessions that walk teachers through the curriculum. During these sessions, teachers focus on both content and associated pedagogical strategies by planning and team-teaching lessons. During the second year, the program provides follow-up classroom support through coaching. Findings indicated that coaching promoted changes in pedagogy, enriched teachers' CS content knowledge, and helped break CS teacher isolation in schools (Margolis et al., 2017).

While the ECS PD program focuses on teachers responsible for teaching a standalone CS course, studies focusing on teachers interested in integrating CS content into existing course curricula are sparse. Recently, Fancsali et al. (2020) explored teacher outcomes across multiple PD opportunities for K-12 teachers in a large urban district, which included a diverse selection of year-long programs focusing on foundational and advanced CS curricula, lasting anywhere between 48 and 100 h. These opportunities were designed to prepare teachers to lead CS courses (grades 6–12) or integrate CS units into other courses (grades K-8). Findings indicated that teachers rated the PD highly and 70–75% of them implemented CS in their classrooms. Higher implementation rates were reported for standalone courses compared to integrated units. Findings also revealed differences among teachers serving students with high economic needs and lower academic performance compared to teachers serving students with lower economic needs and higher academic performance.

Focusing specifically on primary school teachers, Kong et al. (2020) examined the design and delivery of a PD program extending over 39 h, which included two courses focusing on content knowledge for programming, app development, teaching in the classroom, and reflection. Findings indicated that participating teachers noted improvements in their understanding of programming while the opportunity to acquire teaching experience and reflect on practice where fundamental to teacher learning. Similarly, Rich et al. (2021) examined teacher outcomes following participation in continuous PD associated with the BootUp K-8 program (https://bootuppd.org/professional-development/). The BootUp PD program extended over one year and focused on coding and computational thinking, two key aspects of CS. Results indicated that participation increased teachers' confidence, self-efficacy, and comfort level with coding and computational thinking. Albeit valuable, these studies only focused on one aspect of CS, namely programming or coding, and relied heavily on surveys with less attention on other data sources (e.g., Kong et al., 2020). In this work we move beyond programming and use multiple data sources to triangulate findings.

3. Theoretical framework

Identifying the knowledge needed to teach CS requires theoretical models for organizing, presenting, and assessing teacher learning. Traditionally, teacher knowledge has been distinguished into three types: content knowledge (CK), pedagogical knowledge (PK), and pedagogical content knowledge (PCK; Shulman, 1986). Building upon Shulman's work, the TPACK framework (Mishra & Koehler, 2006) focused specifically on how technology (TK) can be integrated with CK and PK to support effective use of technology in teaching.

Despite the wide use of TPACK, its application and research in the domain of CS has been limited. Yadav et al. (2016), for instance, developed teaching vignettes to measure teacher knowledge and understanding of programming constructs with the objective of developing a validating measure for CS teachers' PCK. Similarly, Giannakos et al. (2015) measured secondary CS teachers' knowledge regarding the components of TPACK and mapped those to specific profiles. Findings indicated strengths in CK, PK, and TK, but identified needs in transforming and applying CK into teaching practice. It is important to note that this study was situated in an

Table 1
TPACK in Relation to CS Guiding this Work.

	Content	Pedagogy	Technology
Definition	The big ideas of CS (College Board, 2017)	Knowledge of general pedagogical strategies, as well as knowledge specific to the teaching of CS	Encompasses both general tools, as well as CS specific tools and resources (Vivian & Falkner, 2019)
Examples	Algorithms	*General PK*	*General TK*
	Programming	Assessment	Hardware
	Abstraction	Scaffolding	Software
	Creativity	Curricular connections	
	Data		
	Internet	*CS-Specific PK*	*CS-Specific TK*
	Impacts of CS	CS Unplugged[a]	Scratch
		Pair programming[b]	HTML programming
		POGIL[c]	Data visualization tools
		Adaptation of resources	Robotics
		Broadening participation[d]	Code.org tutorials
			CS Unplugged repository

[a] Kinesthetic activities that teach CS concepts without computers (Bell et al., 2009).
[b] A technique where two programmers work together at the same station.
[c] Process Oriented Guided Inquiry Learning (POGIL): Activities that engage students in active construction of learning while working in small teams (see https://pogil.org).
[d] Specific strategies for broadening participation in computing.

educational context where CS teachers are required to have a computing degree, which is not the case in the U.S. and other countries (Sentance & Humphreys, 2018). More recently, Vivian and Falkner (2019) conceptualized TPACK for CS focusing explicitly on the domains of content, pedagogy, and technology. Building upon the work of Vivian and Falkner (2019) as well as our own prior work (Codding et al., 2021; Pollock et al., 2017), Table 1 indicates how teacher knowledge was conceptualized and applied in the context of the current study.

4. Context of this work: description of the PD program

The PD program examined in this work includes two key components: a face-to-face week-long summer institute and follow-up classroom support during the academic year provided by university undergraduates with knowledge of CS. The summer institute is organized into two tracks: (a) the *Module Track,* which prepares teachers in grades 5–12 to integrate CS principles into existing STEM modules, and (b) the *Course Track*, which focuses on middle or high school teachers responsible for the implementation of CS curricula through standalone courses, specifically the Advanced Placement CS Principles course. Table 2 presents the PD program in relation to key principles of effective PD.

As noted, the week-long summer institute is organized around two tracks that correspond directly to the needs of participating teachers. Table 3 presents a snapshot of the summer institute for both tracks.

Our university service-learning course provides ongoing support throughout the academic year, with undergraduates directly supporting teachers in their classrooms (Pollock et al., 2015). The course, *Field Experience in Teaching Computer Science*, which is open to university undergraduates who have completed at least one prior CS course, combines college classroom meetings with classroom visits. College meetings are devoted to identifying and implementing CS teaching resources targeted toward CS principles, modeling classroom lessons, discussing teaching pedagogy, preparing and analyzing lesson plans, and reflecting on the field experiences (Pollock et al., 2015, Mouza et al., 2016). Field experiences take place in schools where PD participants work to apply what they learned. Throughout the duration of one semester, undergraduates work collaboratively with teachers to adapt lessons and activities from available resources, lead classroom sessions, or support students as teachers deliver instruction (Mouza et al., 2016, 2020).

Typically, each teacher is partnered with 2–5 undergraduates and spends approximately 3 h per week in the field. Due to limited capacity and travel constraints, we are not able to support all teachers who request follow-up classroom support. Teachers are selected based on proximity to the university campus, scheduling availability of undergraduates, and demographics of the student population served, with priority given to teachers in schools serving a greater number of students under-represented in CS.

5. Methods

5.1. Summer institute participants

Participants included all teachers who attended the summer institute over a 3-year period and completed data collection instruments (N = 94). All teachers taught in a Mid-Atlantic state. Of those, thirty-three (33%) were male and sixty-seven (67%) were

Table 2
Description of professional development program.

PD Principles	PD Activities	Expected Outcomes
Content Focus	a. Hands-on activities built around 7 key CS principles or the CSP curriculum b. Pedagogical strategies for teaching CS (e.g., pair programming, CS unplugged, assessment of computational artifacts, etc.) c. Broadening participation in computing activities (designing interactive exhibits, readings/discussions focusing on diversity, strategies for differentiating instruction, etc.)	Build teacher knowledge of key CS principles Build teacher pedagogical knowledge for teaching CS Build teacher knowledge for recruiting diverse student populations in CS and responding to diverse student needs
Active Learning	a. Instructors modeled activities that teachers could enact with their students b. Teachers designed lessons that integrated CS principles with curricular standards (Module Track) c. Instructors/peers provided feedback on lesson design d. Team-based projects e. Time for sharing, discussion, and reflection	Foster meaningful learning: Facilitate application of new learning around CS principles into the classroom
Collective Participation	a. When possible engaged groups of teachers from the same school or groups of teachers from the same content area or grade level	Foster the development of professional communities
Coherence	a. Teachers designed lesson plans that integrated CS principles with curricular standards (Module Track) b. Teachers took turns leading lessons from the CSP curriculum they were expected to teach back in their classrooms (Course Track)	Maintain coherence with school expectations and facilitate application of learning into classroom practice
Sufficient Duration	a. Activities enacted over an extended duration: summer institute & follow-up classroom support.	Provide space to develop/apply new learning into practice and receive feedback.

Table 3
Snapshot of week-long summer institute.

	CS Content	CS Pedagogy/Activities	
		Module Track	*Course Track*
Day 1	Algorithms: algorithm, program, control structures; algorithm examples	a. Generating activities that require the using algorithms: baking, solving puzzles, putting together IKEA furniture b. Generating algorithms: solving a maze c. CS unplugged: Harold the Robot	
	Programming	d. Hour of Code e. Object Oriented Programming Activities (3 different levels)	
	Impacts of Computing	f. Reading Assignments: Impacts of Computing	
Day 2	Programming: sorting strategies, flowcharts Impacts of Computing	a. Writing code that uses different sorting methods b. Reviewing lessons that utilize algorithms and programming and mapping to content area standards c. Programming an adventure game using different strategies: Pair programming d. CS unplugged: Battleships – Searching Algorithms e. Creating interactive programming exhibits for student recruitment; POGIL	Hardware: choosing computer hardware and developing a presentation Solving problems with different types of algorithms
Day 3	Abstraction: definition, data visualization	a. Exploring data; reflecting on visualizations; Generating questions based on visualizations	Levels of abstraction; Constructing computational models from data
	Data: binary representations of different kinds of data, findings patterns in large data, building models of data	b. Group exercise by working with one of the following data sets: US Geological survey seism activity data set (numerical data), Titanic Data (graphing exercise), Donner Party Data (sorting & answering questions) c. CS unplugged: Binary Numbers Tales from the field: Participants share implementation stories	Exploring binary and continuous data Experiencing lessons from computing curricula as learners (Lesson: Encoding images)
Day 4	Internet	a. A simulated routing activity b. CS unplugged: Text Compression	Advanced programming and web applications
	Creativity	c. Examining rubrics for assessing creativity; generating criteria for assessing creativity	Review of AP curriculum and creating performance tasks for students Curriculum pacing, adaptation, planning, creativity
	Impacts of Computing	d. Recruiting for Diversity Strategies: Discussing differentiation strategies for students with special needs, EPR simulated exercise	
Day 5		a. Lesson Planning/Presentation of Lessons/Implementation Plans	

female. Table 4 presents the number of teachers who attended the summer institute and the PD track they completed. While only one teacher had a formal background in CS, all teachers in the *Course Track* had a background in a related STEM field (e.g., mathematics). Teachers in the *Module Track* had a wide variety of backgrounds. Similar to other studies, due to the diverse range of teacher backgrounds, it is not possible to provide concrete descriptions of what a "typical teacher looks like in this context" (Sentance & Humphreys, 2018, p. 351). Participants attended the summer institute voluntarily and received financial incentives for participation.

5.2. Case study participants

At the end of the 3-year period, we contacted all teachers who had participated in a summer institute and requested an interview to examine how they had applied their learning into practice. Of those, 10 teachers moved schools and could not be contacted. Of the remaining 84 requests, 28 teachers agreed to be interviewed. Of those, 8 teachers utilized all components of our PD program: week-long summer institute and follow-up classroom support for at least one semester. Most participants received follow-up support for more than one semester. Here, we focus on a more detailed examination of the practices of the 8 teachers who utilized all components of the PD. Focusing on those teachers was important for identifying how specific PD design features influenced teacher learning and application of learning in practice beyond the summer institute. Table 5 illustrates the characteristics of the 8 case study participants.

5.3. Data collection

Survey Data. A pre/post survey was administered to all participants (N = 94) during each summer institute. The survey was developed by a group of evaluators of National Science Foundation funded projects focusing on preparing teachers for CS teaching. The survey uses a 5-point Likert-scale response format that ranges from 1 (strongly disagree) to 5 (strongly agree) to examine teachers' self-reported CK for CS related to the seven big ideas of computing: creativity, abstraction, data, algorithms, programming, Internet, impacts of computing. It also uses a 4-point Likert-scale format to examine teachers' self-reported knowledge and confidence in pedagogy for CS. Reliability tests yielded high internal consistency for the survey (an alpha coefficient of ≥ 0.70) (Nunnally, 1978). Cronbach's alphas for the seven content and ten pedagogy items were 0.89 and 0.92, respectively. In addition to Likert-scale items, we added two open-ended questions. The first question asked participants to explain how technology tools can be used to solve real-world problems. The second question asked participants to share their perspective on the PD activities of the summer institute. This question was only included in the post-survey.

Interview Data. Interview data were collected from 8 teachers who participated in the full PD program – summer institute and follow-up classroom support. The interviews were conducted by professionals from an education research center, with no PD organizers present, which allowed teachers to express their opinions freely. The interview protocol included 14 questions that focused on: (a) teachers' involvement in the PD; (b) specific CS content, pedagogy, and technology skills learned during the PD; (c) application of new learning in the classroom; (d) contextual factors that facilitated or impeded application of new learning into practice; and (e) ways in which students responded to CS instruction (Pollock et al., 2017). Interviews were conducted face-to-face or electronically (e.g., phone). The interviews typically lasted 30–45 min. All interviews were fully transcribed for analysis.

Classroom Implementation Data. Classroom implementation data were collected through weekly reflections completed by CS undergraduates in the field experience course. An important component of service learning is directed reflection that connects what undergraduates learn in the classroom with their experience of real-world conditions (Bringle & Hatcher, 1999). Every week, undergraduates reflected on their experience using three prompts: (a) field activities enacted during the week, (b) identified successes and failures, and (c) recommended steps for future action (Pollock et al., 2015, Mouza et al., 2016, 2020). Reflections were completed as homework outside of structured class time. A total of 588 reflective entries were collected from 49 undergraduates who worked with the 8 case study participants during the 3-year period. The length of each entry varied, with an average of 400 words per entry. When compiled, reflection entries totaled 477 single-spaced pages.

5.4. Data analysis

Survey Data. Quantitative survey data were analyzed using the Wilcoxon Signed-Ranks test (Field, 2013), a nonparametric test for paired data to examine gains in teacher knowledge of CS content and pedagogy following their participation in the summer institute. Open-ended survey responses were analyzed qualitatively with a focus on examining how participation in the summer institute influenced teacher learning, specifically in relation to technology tools and their application in real-world settings. Our analytical approach was inspired by grounded theory (Glaser & Strauss, 1967). We utilized open coding to develop a coding scheme from emergent themes based on the individual survey responses (Strauss & Corbin, 1990). The coding scheme was subsequently applied and refined in two rounds of coding. Table 6 provides excerpts from the final code book. Using Dedoose we coded 898 excerpts with 1544 code applications.

Table 4
Summer institute participants (N = 94).

	Year 1	Year 2	Year 3
Module Track	24	16	8
Course Track	13	19	14

Table 5
Case study participants.

Name	Year of Participation	PD Track	School Level	Subject Area	No. of Semesters Received Support	No. of Undergraduates Provided Support[a]
Jamie	Year 1	Module	High School	Content Teacher	1	3
Karen	Year 1	Course	Elementary & Middle School	Technology-CS Teacher	3	9
Brandon	Years 1–2	Course	High School	CS Teacher	2	9
Victoria[b]	Years 1–3	Module	Upper Elementary	Technology Teacher	7	22
Jasmine	Year 2	Module	Upper Elementary	Content & Technology Teacher	2	6
Ross	Year 2	Course	High School	CS Teacher	2	3
Martha	Year 3	Module	Middle School	Technology Teacher	2	13
Shirley	Year 3	Module	Elementary & Middle School	Technology Teacher	1	4

[a] Some undergraduates worked with more than one teacher and thus numbers do not round up to 49 – total number of undergraduate participants.
[b] Following her participation in the summer institute, Victoria established an after-school computing program ran every semester and open to any interested student in her school. The program was delivered in collaboration with groups of undergraduate facilitators.

Interview Data. Qualitative methodologies were chosen for this study to "illuminate the meanings people attach to their words and actions," which would not have been possible without interviews that provided essential insight into teachers' experiences during and following participation in the summer institute (Lareau, 1996). Interview data were analyzed using an inductive strategy inspired by grounded theory (Corbin & Strauss, 2015; Glaser & Strauss, 1967). Codes were developed, refined, and applied during three rounds of coding by one researcher (see Table 7). Qualitative analysis allowed us to identify common and unique themes that were then organized into two overarching categories: application of PD learning into practice, and benefits of situated classroom support provided by university undergraduates.

Classroom Implementation Data. Reflective journal entries collected from undergraduates were used to triangulate teacher interview data and gain a better understanding of classroom implementation. Journal entries were grouped by corresponding case study participant (see Table 5). The final codebook (see Table 7) was then applied to reflective journal entries during two rounds of coding by one researcher. Additionally, qualitative data from teacher interviews and reflective journals were used to construct case narratives, which were organized into a framework that included a description of the teacher and their K-12 school, an examination of how the teacher applied CS content and pedagogy in the classroom, and a discussion of PD design features that influenced their teaching (Merriam & Tisdell, 2016; Yin, 2018).

The 8 case studies provide a detailed portrait of each participating teacher, which allowed us to examine each teacher's interview

Table 6
Excerpts from code book.

Parent Code	Child Code	Definition	Example
Use of Technology in Response to Societal Problems		Teacher identifies a way in which technology is used to solve a specific societal problem	You can use lasers for eye surgery instead of needles. You can drive trucks via remote, first one was released in AZ this year.
	Data	Problems concerning the collection, analysis & encryption of data	Providing for a way to search through the vast amount of data which has become available.
	Science & Society	Problems concerning issues of science and society	Create apps to communicate with doctors and amass states on symptoms, in real time, for patients with diseases like Sickle Cell.
	Banking & Finance	Problem concerning issues connected to banking & finance	Catching bank fraud through signature recognition.
	Impacts of Computing	Identify ways in which the students can use technology to solve a social or community problem	My students built a parking application for our school district that registers student vehicles and assigns parking space. It also handles tickets and fines for students and warnings for visitors and staff.
Technology as Tool		Use of technology as a tool to accomplish something, without a focus on creating, programming, or problem solving	Students can graph parabolic functions on their calculator to determine height at which objects are thrown, how long it takes for them to hit the ground, etc.
	Research & Self-Help	Technology as a preexisting tool that we use to help ourselves in our daily lives	When students are facing challenges in the classroom, they can use technology to help them through it.
	Connection, Collaboration & Communication	Technology as tool to connect, collaborate & communicate via the internet	One particular way that technology can be used to solve real-world problems is through its ability to let different people collaborate and share ideas.
	Increased Efficiency	Technology as tool for increasing efficiency, frequently applied to issues of scheduling & transportation	Create more efficient routes for travel.
Vague Connection		Teacher fails to articulate a clear/coherent connection to a real-world application of CS/CT	Algorithms, Programming. Most problems can be solved with assistance from technology.

Table 7
Evolution of themes and code refinement.

Guiding Research Questions					
1. How did participation in the summer institute influence teacher learning of CS content, pedagogy, and technology? 2. In what ways did teachers who received follow-up classroom support implement new learning into CS practice? 3. How did specific PD design features influence teacher learning and implementation of CS content, pedagogy, and technology back in their classrooms?					
Open Coding					
• Creativity • Student Excitement/Enthusiasm • Undergrad Student Support • Increased engagement with undergraduate students' support • New/uncomfortable with content Teacher as resource	• Teacher as expert • Filling a need in the district • Limited time • Teacher learning alongside students • Fear of not knowing • Training as example, not template	• Needs more explicit template to follow • Brought about change at their school • Bringing in diverse students • Connecting CS to "real life" and other subjects • CS as exposure for students			
First Iteration Codebook					
Themes					
1. Outcomes from Field Experience	2. Limitations Faced by Teachers	3. Implementing PD Resources	4. Role of CS Teacher	5. How Teachers Frame Their Success	6. Advice for Other Teachers
Codes					
1A Near-peer mentorship for students 1B Contextualized support 1C Teacher comfort with CS content and pedagogy 1D Extra hands 1E Student engagement 1F Adaptation and problem solving 1G Scheduling & planning problems	2A Lack of curriculum 2B Limited resources 2C School expectations 2D Difficulty with differentiation/ scaffolding 2E Lack of content knowledge/limited understanding of CS principles	3A CS principles 3B Resources 3C Teacher confidence 3D Resources (indirectly used) 3E Pedagogy – paired programming 3F Student focus/ student engagement	4A Learning *with* students 4B CS expert 4C Resource 4D Teachers set the tone of the classroom	5A Student autonomy 5B Increased interest 5C Expanding access (broadening participation)	6A 'Just do it' 6B Turn to online resources 6C Let the kids go for it 6D Do it for the kids
Second Iteration Codebook					
Themes					
1. Applications of PD Learning into Practice			2. Benefits of Situated Classroom Support		
Codes					
1A Teaching Strategies and Approaches (Pedagogy) 1B CS Principles (Content) 1C CS Resources & Tools (Technology) 1D Challenges/Limitations			2A Near-Peer Mentoring 2B Hands-On Support 2C Increased Student Engagement 2D Adaptation And Problem Solving		

more holistically, triangulate data through the critical analysis of weekly undergraduate reflection journals, and perform cross-case analysis. We used a case-based approach to cross-case synthesis to retain the integrity of each case and to compare within-case patterns across the 8 cases (Yin, 2018). To ensure trustworthiness, we used several qualitative strategies for promoting validity and reliability, such as triangulating findings through multiple data sources, transcribing audio recordings and manually cleaning transcripts, and engaging with data collection over a 3-year period (Creswell, 2007; Merriam & Tisdell, 2016).

Table 8
Distributional statistics for self-reported knowledge of CS principles before and after PD.

CS Principle	Median Rank (Pre)	Median Rank (Post)	Z Statistics	Effect Size (d)
Creativity	3.00	4.00	−6.32	−2.00
Abstraction	2.50	4.00	−6.56	−2.16
Data	3.00	4.00	−6.09	−1.86
Algorithms	2.00	4.00	−6.81	−2.35
Programming	2.00	4.00	−6.80	−2.34
Internet	4.00	4.00	−4.88	−1.30
Impacts	3.00	4.00	−6.46	−2.09

Note: $N = 80$ (Although 94 participants completed a pre and/or post survey, only 80 of them were fully completed and included in the analysis).

6. Results

6.1. Teacher learning of CS content, pedagogy, and technology

Analysis of survey data indicated that the summer institute positively influenced teacher learning of CS content, pedagogy, and technology. Table 8 presents ranks for the pre and post survey responses on the dependent variables of seven CS principles. Results from the Wilcoxon Signed-Ranks test showed statistically significant differences ($p < .001$) on teachers' self-assessed knowledge of CS principles from the pre to the post administration of the survey. The obtained differences represent large effect sizes (Cohen's, 1988, $d > 0.80$).

Table 9 presents ranks for the pre and post survey responses on the dependent variables of pedagogy for teaching CS. Results from the Wilcoxon Signed-Ranks test showed statistically significant differences ($p < .001$) on teachers' self-assessed knowledge of CS pedagogy from the pre to the post administration of the survey. The obtained differences represent large effect sizes (Cohen's, 1988, $d > 0.80$).

When asked to provide two specific examples of how technology tools can be used to solve real-world problems, pre-survey responses revealed two broad themes: uses of technology in response to societal issues, and applications of technology as a tool to accomplish everyday tasks (Table 10). Specifically, approximately 45% of responses provided examples of technological applications drawing from areas such as data (17%), science and society (17%), banking and finance (6%), and impacts on communities (5%). Most examples focused on issues of data (e.g., searching large datasets) and the role of computing in science such as solar power, human genome, and homeland security. Notably, these were typically addressed in single words or phrases, indicating that participants understood the connections between technology (general and CS-specific) and society but lacked the content knowledge to make such connections explicit. A smaller number of responses focused on describing applications of computing in banking and communities. For instance, participants recognized that "technology can help organize personal finances" and "build secure banking" (Year 3). Similarly, addressing the impacts of computing, a participant indicated that students had "built a parking app for our school district that registers student vehicles and assigns parking space" (Year 1).

Pre-survey responses also indicated a focus on the use of technology as a tool to support research (30%), communication, and collaboration (8%). Specifically, teachers recognized that technology can be used for research and assistance with daily tasks, such as "find a quicker way to a destination," "differentiate instruction," "assist students with disabilities," and help all students become "self-directed and independent learners". Notably, these examples focused on individuals as *users* of technology and left little room for invention or unique applications. Finally, a smaller number of responses emphasized the role of technology to support collaboration and social change. As one participant explained, "tweets can be used in a conflict zone to pre-direct emergency responders" (Year 2). In these examples, existing types of technologies were presented as tools for achieving solutions to real-world problems.

Post-survey responses mirrored the two broad themes identified in pre-surveys– uses of technology for societal issues and technology as a tool. Specifically, more than half of the responses (53%) provided examples on the use of technology in response to societal issues, particularly in relation to data and the role of technology in science. However, teachers' examples were detailed and complex compared to the pre-survey responses. Describing the role of technology in relation to data, one participant recognized the role of technology in "collecting and interpreting data about natural disasters to make predictions" (Year 3). Other participants noted the role of data in the medical field, explaining that "health data can be collected and analyzed for patterns to avoid potential epidemics" (Year 2). Discussing the role of technology in science and society broadly, participants also provided a range of applications focusing on advances such as the role of robots in performing jobs too dangerous for humans and self-driving cars. Finally, a number of responses addressed impacts on communities (10%), such as using computing to "map out more efficient bus routes" (Year 2).

Post-survey responses also indicated that participants continued to view technology as a tool to support research (14%), communication and collaboration (7%), and efficiency (10%). Regarding efficiencies, many participants provided examples from their daily work with students, such as automating procedures, sharing documents, and creating school schedules. Describing the role of technology in supporting communication and collaboration, one participant explained: "Technology serves to level the playing field, whereas, in the past information was the purview of the wealthy and privileged. Now the masses have access to this information and can use it to stay informed and compete for the occupations of the future" (Year 2). Another participant took this idea a step further

Table 9
Distributional statistics for self-reported pedagogical knowledge in teaching CS before and after PD.

CS Teaching Statement	Median Rank (Pre)	Median Rank (Post)	Z Statistics	Effect Size (d)
Plan Differentiating Instruction	3.00	3.00	−4.32	−1.10
Teach Computing to Students with Learning Disabilities	2.00	3.00	−3.71	−0.91
Teach Computing to Students with Physical Disabilities	2.00	3.00	−4.12	−1.04
Teach Computing to English Language Learners	2.00	3.00	−4.31	−1.10
Provide Enrichment Opportunities for Gifted Students	3.00	4.00	−4.24	−1.08
Encourage Student Interest in Computing	3.00	4.00	−6.14	−1.89
Teach Computing to Girls	3.00	4.00	−4.43	−1.14
Teach computing to Ethnic Minority Students	3.00	3.50	−4.37	−1.12
Teach Computing to Low SES Students	3.00	3.00	−4.00	−1.00
Teach Students the Relevance of Computing in Daily Life	3.00	4.00	−5.47	−1.55

Note: $N = 80$ (Although 94 participants completed a pre and/or post survey, only 80 of them were fully completed and included in the analysis).

Table 10
Teacher Perceptions of How Technology Can be Used to Solve Real-World Problems.

Parent Code	Child Code	Pre-Survey	Post-Survey
Use of Technology in Response to Societal Problems		45 (45%)	59 (53%)
	Data	17 (17%)	24 (21%)
	Science & Society	17 (17%)	22 (20%)
	Banking & Finance	6 (6%)	3 (3%)
	Impacts of Computing	5 (5%)	10 (9%)
Technology as Tool		39 (39%)	35 (31%)
	Research & Self-Help	30 (30%)	16 (14%)
	Connection, Collaboration & Communication	8 (8%)	8 (7%)
	Increased Efficiency	–	11(10%)
Vague Connection		17 (16%)	18 (16%)

noting how communication technology can "link people and ideas together and help change the world" (Year 2).

6.2. Application of new learning into practice

Interview and classroom level data indicated that teachers were able to apply new learning into practice by adapting and utilizing curricular resources, technology tools, and pedagogical strategies shared in the PD. Classroom implementation was driven by students' interests and shaped by teachers' comfort level with CS.

6.2.1. Implementation of CS principles (CS content)

As noted, the content of the summer institute was organized around big ideas of CS, called CS principles. Following their participation in the summer institute, teachers implemented CS principles in varied ways. Some teachers focused on implementing all CS principles into their curriculum as illustrated by Brandon: "I think they're all important and they all play a role in the decisions my students will be making as they become practitioners in the field." Brandon used the CS principles as the framework for his curriculum. Similarly, Ross used the PD as a "framework to build off of" while developing a CS curriculum for his school that integrated all CS principles. Other teachers based their content choices on students' interests and engagement. For Victoria, the integration of CS principles "depended in part on what kinds of projects the students want to work on." Victoria also shared with the undergraduate facilitators that it was "critical that they make examples and analogies in terms that her students could understand" (*Undergraduate Reflection, Year 1*). Nonetheless, five of the teachers noted that they found it challenging to integrate CS principles with core curricula. Victoria explained, "It's really about how can I work it in around what is required of me in terms of the school district and the school's curriculum." Victoria tried to integrate CS principles in subject areas outside CS, but she was limited by her curriculum requirements. Other teachers used CS principles to guide their curriculum choices, such as introducing programming and data in mathematics and science.

Increased confidence in CS content was important in helping teachers implement CS principles. Teachers focused on the CS principles they were comfortable with, as noted by Karen who focused primarily on programming because "it's definitely the more concrete of the principles and one that I'm more comfortable with." Consistently, however, teachers identified creativity as the most engaging principle. Martha, Jasmine, and Jamie, for instance, encouraged creativity through programming. Martha explained: "Students are creating backgrounds, working on story telling through Scratch, and making a live graphic novel." When asked about other CS principles, Jamie noted, "I don't know if I understand them thoroughly yet." This lack of understanding limited the breadth of CS content integrated in her classroom. Jamie's students primarily experienced CS through programming, which, as she noted, created a struggle in terms of maintaining their interest. Similarly, Ross noted that he struggled with algorithms and abstraction and as a result his students struggled with them as well.

6.2.2. Implementation of CS pedagogy and technology

Most teachers were selective in the teaching strategies they implemented and how they adapted them for their classroom context. Teachers found the summer institute empowering in this regard. As Martha noted, "It was great, especially for someone who doesn't necessarily have a CS background. It empowered me." Findings indicated that teacher confidence influenced choice regarding which teaching strategies to utilize in the classroom. Due to their limited content knowledge, both Karen and Shirley often relied on pre-existing online tutorials for students. Specifically, students followed along with the block coding tutorials available on two online platforms, Scratch and Code.org: "They would watch it and then do it … like me." Additionally, Shirley expressed concern with her inability to solve CS problems for her students. When asked about her greatest challenge, Shirley said, "I don't have a strong aptitude for it myself, and when the kids get to a certain point, I try not to get in their way but when they need help, I'm stuck." Other teachers, like Jasmine, however, were not afraid to learn alongside students: "So many times, the kids and I sit together and problem-solve. Sometimes they find an answer, sometimes I do."

CS Unplugged. Six of the teachers utilized kinesthetic activities, called CS unplugged, introduced during the summer institute. Jasmine, for example, used kinesthetic activities with almost every lesson. Other teachers appreciated that CS unplugged provided a reliable way of teaching CS especially when they encountered technological or network challenges. Victoria explained: "They have been really fabulous for days when I come in and all of our computers are down." However, Brandon found that his high school

students did not enjoy these type of learning activities, as they were not perceived as CS. Further, Karen found that she had to justify the use of such activities to her students and parents. She explained: "There's been kind of a learning curve for the kids and their parents to understand that you can still be doing computer science even if you're not touching a computer at the time."

Pair Programming. Pair programming was a new strategy for most teachers. Following the summer institute, seven teachers tried to implement this strategy into their classrooms. Karen implemented pair programming because it pushed her to "let go of control" and let students teach each other. The undergraduate facilitators observed Karen's students "taking turns controlling the computer … learning to vocalize their ideas in a way that others can identify and follow" (*Undergraduate Reflection, Year 1*). The undergraduate facilitators also placed an emphasis on pair programming as they co-planned activities with the classroom teachers, having experienced it in their own learning at the university. As one undergraduate facilitator explained: "Having two sets of eyes helps you to catch problems that you would otherwise miss and brings a new perspective to the game in times of confusion and bugs" (*Undergraduate Reflection, Year 3*).

While most teachers successfully implemented pair programming, two teachers shied away from this new strategy. Shirley did not implement pair programming because her students "have a ton of technology, so they don't have to share." Because Shirley viewed pair programming as a strategy that should only be done out of necessity, she failed to incorporate it. Ross also found pair programming challenging because he had to "relinquish power to the kids" and "just facilitate." However, he also noted that pair programming "was a strategy [he] ended up enjoying a lot."

Process Oriented Guided Inquiry Learning (POGIL). Five teachers noted they found it challenging to keep up with the latest technologies, which impacted their pedagogical choices. Brandon addressed this problem by using POGIL exercises. Brandon put his students to work "learning through the problems," because he thought it was important for them to "hit some walls, make some errors, and learn from those mistakes themselves." However, not all teachers were as confident with their ability to learn alongside the students. Jamie, for instance, described herself as "not a computer science person," even after attending the summer institute. The undergraduates also noticed her lack of confidence in CS: "She was great, seemed to have a good relationship with her kids, very friendly, but was extremely novice at programming, and I think computer science in general" (*Undergraduate Reflection, Year 2*). Several teachers expressed concern with their inability to serve as facilitators of students' CS learning and help solve problems for their students. Other teachers developed modified versions of the teaching strategies to better fit their context. Jasmine, for example, used a modified version of POGIL where she gave her students parameters and they had to program within those parameters. During one POGIL project, her students created recycling games for Earth Day: "They had to conduct research, and I didn't give them strict guidelines. I gave them light parameters and then they had to ask the questions themselves and find the answers."

Adapting Curricular Resources. The summer institute provided teachers with a collection of resources. However, teachers discussed the need to adapt the materials for their specific context to address their student needs. Victoria explained: "[I] pretty much adapted [the summer institute modeled activities], because when they were presented to us, they were presented for older students … Not just in terms of their programming ability, but perhaps more importantly, their logic and understanding, and their math ability and those kinds of things depending on the project." Like Victoria, several elementary and middle school teachers perceived the summer institute as oriented primarily toward higher grade teachers.

Teachers also adapted curriculum resources in response to student interests and current events. Several teachers noted that they avoided preexisting curricula "mainly because the kids got bored with them. They weren't able to express themselves, they weren't able to build what they wanted to build, and they got frustrated with that" (*Jasmine*). Additionally, Brandon found that advances in technology often left textbooks and online curricula outdated. While discussing ethical issues in CS, Brandon and his students "pulled down real-time events happening in the news and studied that instead of going back to the book." On another occasion, "The students asked a lot of questions concerning advanced functionality of websites and were eager to know how many things worked. One student's question led Brandon and I on a semi-tangent about web servers and the architecture of the Internet" (*Undergraduate Reflection, Year 2*). Brandon's content knowledge and personal interest in CS allowed him to quickly adapt his teaching in response to student questions and interests.

Although some modifications were made to better suit the students, others were made to better suit the teacher. Limits in CS content and technology knowledge, for instance, made it difficult for some teachers to use specific program elements. Shirley limited her class to Scratch, as she was "a little apprehensive" to move into tools and resources she did not understand. Teacher interest and passion also drove the inclusion of technology and tools. Jasmine, who had a background in engineering and technology, used a lot of different technologies in her classroom and was always thinking up new ways to support her students: "I do whatever I can do to help the kids learn." Jasmine's passion for teaching CS set the tone for her students. As one undergraduate facilitator noted, "kids respond very well to you when it is clear that you are passionate about what you are doing" (*Undergraduate Reflection, Year 2*).

6.3. PD design features supporting teacher learning and practice

Results indicated that the summer institute positively impacted teacher learning but undergraduate facilitators were instrumental to the implementation of new learning into practice.

6.3.1. Summer institute

Post-survey responses collected from all teacher participants (N = 94) indicated two elements primarily valued by summer institute participants: focus on content and pedagogical strategies for teaching CS (n = 47), and community building (n = 32). Overall, teachers appreciated the opportunity to deepen their content knowledge (CK). As most teachers were new to CS, they needed to gain an overall understanding of CS principles first. As one teacher noted: "I think the breadth of the PD was the best part" (Year 2). The focus on content empowered teachers, helped take away the "fear" of teaching CS, and fostered a sense of confidence in their ability to teach CS.

One teacher explained, "I feel much more confident in my own understanding of CS. I was always afraid of it, and now I realize I can learn the concepts and build skills. I always thought it was only for math and science wizards, but that myth has been dispelled" (Year 2). The resources and materials shared during the summer institute were also valued by teachers because they were highly accessible.

Additionally, the majority of teachers appreciated the CS pedagogical strategies modeled during the summer institute. As one teacher explained, instructors guided teachers through "creative ways to introduce complex CS concepts to students in an engaged manner" (Year 3). This approach helped teachers "learn ways to adjust [their] instruction for different learners" (Year 3). Importantly, modeling of pedagogical strategies helped participants deepen their understanding of differentiated CS instruction. One participant explained: "The instructors demonstrated exemplary teaching practices … different participants struggled at times with concepts being taught, so we were able to see differentiated instruction for each task" (Year 2).

Finally, most teachers valued the opportunity to meet other CS teachers across districts and schools. Throughout the week, teachers "shared the space with other novice and experts, working together" (Year 3) and valued "hearing other teachers' stories and seeing how other schools are addressing CS" (Year 2). Other teachers enjoyed the time and space to work with colleagues from the same school as noted in one of the responses: "We enjoyed collaborating with colleagues and discussing ways to integrate CS Principles into different classrooms and with diverse students" (Year 2). By allowing participants extended periods of time to work through programming together, they were able to create a supportive network of CS teachers. As one teacher emphasized, "The networking alone was worth the time!" (Year 3). This need for networking was driven by the isolation many CS teachers experience as described by a participant: "Often there is only one person in a building who is teaching CS so there is no peer support when we encounter a problem" (Year 2).

When asked to provide suggestions for improvements, an overarching request was more time for participants to collaborate, write lesson plans, and complete projects. One teacher explained: "I wish we had more time to complete a fully working programming project to have as an example. Not only would this be helpful for learning, but it could be a product to show off to administration as well as students to motivate new CS programs" (Year 2). Other teachers requested further differentiation of content to meet individual teacher needs. For some teachers, this meant "going deeper" into the concepts, as they felt that "many activities were basic" (Year 1). Finally, a number of teachers noted the need for additional instructional strategies that help reach all students, including those with disabilities and English language learners.

6.3.2. Follow-up support: field experience

Interview data revealed that the support received from the undergraduate facilitators was critical to the implementation of learning into practice. Specifically, teachers noted four benefits of having undergraduate facilitators assisting in their classrooms: near-peer mentoring, hands-on support, student engagement, and problem-solving (see Fig. 1).

Near-Peer Mentoring. All 8 teachers perceived undergraduates as highly influential in their classrooms and noted the benefit of near-peer mentorship for students. Specifically, teachers noted the importance of engaging with near-peer role models currently studying CS in college and planning to work in the field. As noted, this was especially true for female and minoritized students. Martha explained, "It allowed [my students] access to great role models … It gave them a chance to see themselves in someone who's studying it, interested in it, and is in the field." Similarly, Brandon found that his students were "looking up to those kids as role models for career choices or college choices or even decision choices … I think that there was some mentoring happening." The undergraduates helped to introduce CS as a possible career path for his students. An undergraduate noted in his reflection: "One student was shocked, in a good way, that computer science was its own major/department. He didn't know this was something you could study and make a career out

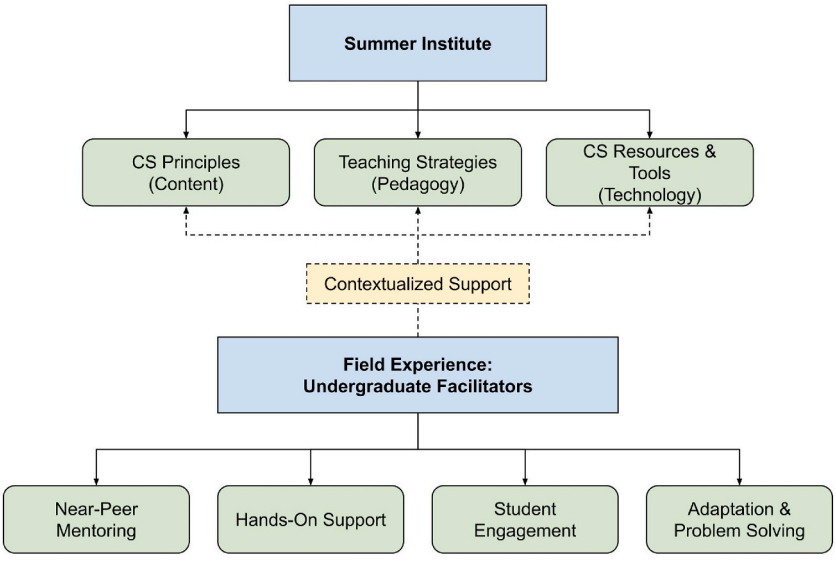


Fig. 1. PD model and associated teacher outcomes.

of' (*Undergraduate Reflection, Year 1*).

Hands-On Support. Undergraduate facilitators were able to assist classroom teachers in a variety of ways, but several were grateful to have an extra pair of hands in the classroom. For Martha, the undergraduate students were "instrumental in helping to solve technology problems and challenges" that she and her students encountered. Additionally, it was "great to have other people to bounce ideas off of … It's great to observe how they're teaching, but also to be able to observe my students doing it." Having the extra weekly support from undergraduates was key to helping teachers implement new initiatives. Several teachers noted that without the help of the undergraduates they would be unable to run their afterschool computing programs. Victoria explained: "Having them here for the after-school club has allowed for very student directed learning. It just wouldn't be possible for me to work with 30 students and really have the kind of student directed learning that we've been able to have." Victoria found that the undergraduate facilitators were energetic and fun, and the students didn't have to wait a long time to get the help they needed. Undergraduates were able to provide knowledgeable, experienced, and qualified CS support in their classrooms. Jasmine reinforced this idea: "It was useful to have people who knew what they were doing there to assist the children with CS learning."

Increased Student Engagement. Undergraduates also brought a level of excitement to their lessons, which helped to engage the students in new activities. Shirley explained after a lesson where the undergraduates demonstrated their own college-level programming work: "The kids were really, really interested because he was creating his own game. They were on the edge of their seat and following along and had a million questions for him." The undergraduate facilitators also emphasized the importance of "making sure to show how passionate you are about a topic because it can really make a difference in how the students react to that subject. I think if [we] didn't show how 'fun' computing was, a lot of the students probably would have lost interest" (*Undergraduate Reflection, Year 3*).

Adaptation and Problem-Solving. A central theme that emerged from the interviews was undergraduates' assistance in adapting and problem-solving in the context of individual classrooms. Typically, teachers met with undergraduates in advance to discuss their needs and determine how new lessons will be integrated into the curriculum. Further, teachers benefited from the individual support and from observing lessons delivered to their own students by the undergraduates. Undergraduates often adapted their lessons in response to student interests: "The students, throughout the semester, were working on their own Scratch projects and many of them were doing things more advanced than my lesson plans. With that being reality, I had to alter my lesson plans, providing only basic features and giving the students time to incorporate them into their own projects" (*Undergraduate Reflection, Year 2*). The undergraduates thus provided a model for responsive adaptation that teachers could implement with future lessons. However, when undergraduates failed to co-plan with the classroom teacher due to scheduling conflicts, their lessons were less effective, detracting from the current curriculum and projects. The undergraduates also found that their activities didn't contribute to the overall class when they did not explicitly plan with the teachers.

7. Discussion & implications

As the teaching of CS gains increased attention across K-12 systems, helping teachers acquire knowledge of CS content, pedagogy, and technology needed for CS instruction has been a tremendous challenge (Qian et al., 2018). This is particularly true for elementary and middle school teachers as the majority have never had to teach CS before (Falkner et al., 2018; Rich et al., 2021). As a result, teacher confidence, pedagogy, PD, and support all need to be addressed (Brown et al., 2014). In this work, we present one approach to PD that helped teachers improve their knowledge of CS content, pedagogy, and technology while providing on-going support for the application of new learning into practice. Findings indicated that this approach to PD holds promise for the successful infusion of CS content in middle and high school classrooms.

7.1. PD and teacher learning

Findings indicated that the summer institute positively influenced knowledge of CS content, pedagogy, and technology. Survey data collected at the beginning of the summer institute indicated that teachers had limited understanding of CS content, particularly in relation to algorithms and programming (see Table 8), concepts that are fundamental to the field of computing (Denning, 2017). The summer institute included a strong focus on algorithms and programming during the first two days, allowing participants to engage in a variety of hands-on activities both individually and with peers through pair-programming. Further, a number of participants incorporated algorithms and programming in their lesson designs following participation in the summer institute.

The principles of data, abstraction, and the Internet were addressed on the third and fourth day of the institute respectively while creativity and impacts were addressed both separately and on multiple occasions throughout the institute as teachers discussed readings and brainstormed lesson ideas. In turn, survey results indicated significant improvements with large effect sizes (see Table 8) across all principles. Additionally, open-ended responses to survey questions indicated a more nuanced understanding of the role of technology, and programming in particular, in solving real-world problems. These findings are consistent with prior literature indicating the need to help teachers build their subject knowledge of CS (e.g., Sadik et al., 2020; Sentance & Humphreys, 2018).

In addition to content, feeling pedagogically confident or having high levels of self-efficacy is also essential to becoming effective teachers of CS (Zhou et al., 2020). Recent work, for instance, indicates that CS pedagogy is the most important aspect of teaching CS effectively (Mouza et al., 2021; Sentance & Humphreys, 2018; Yadav et al., 2016). Survey results indicated that participants significantly improved their pedagogical knowledge in teaching CS with greater gains in four areas: (a) encouraging student interest in computing, (b) teaching students the relevance of computing in daily life, (c) teaching computing to girls, and (d) teaching computing to ethnic minority students. These are all areas substantially addressed and modeled during the summer institute (see Tables 2 and 3).

Survey results indicated that despite improvements, teachers continued to feel less comfortable working with English language

learners and students with disabilities. Arguably, PD activities devoted less attention to strategies for working with students with disabilities and English language learners. Future PD programs can provide additional opportunities for teachers by introducing specific strategies that have been found to support CS learning for students with disabilities including the use of accessibility frameworks, balancing explicit instruction with open-inquiry activities (e.g., POGIL), supporting student-to-student help seeking, and exploring the use of assistive technologies (Israel et al., 2016). Further, future PD opportunities can introduce strategies for working with English language learners, including linguistic scaffolding to help students understand CS technical language and culturally responsive materials that connect CS to students' lives and communities (Jacob et al., 2018).

7.2. Application of new learning into practice

Findings also indicated that case study teachers were able to apply their new learning into practice by adapting and utilizing curricular resources, technology tools, and pedagogical strategies shared in the summer institute. Like content, teachers were selective in which tools and strategies they utilized and how they utilized them. Nonetheless, both CS unplugged activities as well as pair programming and POGIL were implemented by case study teachers. CS unplugged activities were particularly popular with elementary and middle school teachers—such activities provide an easy onramp to infuse CS content into existing curricula (Caeli & Yadav, 2019). Teachers teaching standalone CS curricula, however, were less enthusiastic about CS unplugged based on their students' responses, who did not perceive them to be rigorous CS instruction.

Additionally, findings indicated that teachers rarely implemented pedagogical strategies shared during the PD in uniform ways. Rather, they frequently adapted them to fit their particular context, their own knowledge and comfort level, and their student needs. This finding is consistent with prior research indicating differences in PD uptake based on prior knowledge. For instance, Qian et al. (2018) found that more experienced CS teachers were less likely to implement PD content back in their classrooms while novice and non-computing teachers were more likely to utilize examples and resources shared during the PD. In this work, most participants were novices and thus were more willing to apply content and pedagogy learned in the PD back in their classrooms, but comfort level with content was a mediating factor.

7.3. PD elements valued by teachers

Results indicated that the summer institute alone, albeit useful in supporting teacher learning, may be insufficient to provide the support and preparation needed to teach CS. Rather, continued PD support is crucial, especially for teachers new to computing. Interview data collected from teachers as well as reflections collected from CS undergraduates indicated the important role of situated classroom support. Specifically, by co-planning and co-teaching with teachers, undergraduates were able to impact how teachers made decisions concerning CS content, pedagogy, and technology. This finding is important as we consider best practices in CS PD. Research indicates that the problem-based nature of CS makes it more challenging to plan for lessons that keep students engaged and respond to diverse needs and interests (Sadik et al., 2020). Undergraduates not only supported teacher planning but also assisted teachers in solving context-specific problems and provided a safety net for teachers with limited CS content knowledge.

8. Limitations

There are three main limitations associated with this work. First, all teachers volunteered to attend the PD and work with undergraduate facilitators. As a result, they were already interested in learning and applying CS in their classrooms. Further, participants teaching at the high school level were already expected to teach a standalone CS course in their classrooms. Therefore, it is not surprising that all teachers reported implementing learning from PD back into their classrooms. Second, despite the use of multiple data sources, the study did not include any direct measures, such as classroom observations or performance assessments. Nonetheless, interview data provided in-depth information about teachers' experiences and the broad context surrounding their use of CS. Additionally, undergraduates' weekly reflections provided classroom level implementation data based on their own experiences working alongside teachers. Finally, although the ultimate goal of PD is to influence student outcomes, this work did not collect student data. Future studies should attempt to link participation in PD to changes in teacher learning, classroom practice, and student CS outcomes.

9. Conclusion

As the field of CS is gaining increased attention, the need for qualified teachers is rapidly growing. The number of teachers with the level of understanding and expertise in CS, however, is limited and does not enable large scale implementation of CS (Falkner et al., 2018). This shortage of qualified teachers creates an urgent need for PD initiatives that help participants acquire the content, pedagogy, and technology knowledge they need to integrate CS into existing curricula or teach standalone CS courses. In this work, we examined a PD program for helping teachers in grades 5–12 integrate CS across the curriculum in a variety of content areas or deliver standalone CS courses. The program adhered to high-quality design features reported in the research literature (e.g., Darling--Hammond et al., 2017).

Findings from this work indicated that participants reported improvements in CS content and pedagogical knowledge and applied new learning into their CS practice. Responding on the value of PD design features, teachers noted the importance of focusing on CS content knowledge and opportunities to engage with pedagogical practices for teaching CS. Further, participants highlighted the importance of collaboration and building community with other teachers during the summer institute. Findings, however, revealed

that follow-up classroom support provided by undergraduate facilitators with a background in CS was critical in the implementation of PD learning into practice. These findings suggest that CS-focused PD which adheres to high-quality design features can enhance teacher learning and practice, particularly in relation to foundational CS knowledge. Future PD initiatives, however, need to ensure that as teachers progress in their CS learning and practice, there are continuous opportunities for professional growth.

Credit author statement

Chrystalla Mouza, Conceptualization, Methodology, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Validation, Project administration, Funding acquisition. **Diane Codding**, Methodology, Formal analysis, Data curation, Writing – original draft, Writing – review & editing. **Lori Pollock**, Investigation, Writing – review & editing, Resources, Project administration, Funding acquisition.

Acknowledgement

This work was supported by the National Science Foundation under Award Numbers 1639649 and 1240905. We wish to thank Hui Yang for her assistance with the quantitative survey data analysis.

References

Basu, S., Rutstein, D., & Tate, C. (2021). *Building teacher capacity in K-12 computer science by promoting formative assessment literacy*. National Comprehensive Center.
Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *Journal of Applied Computing and Information Technology, 123*(1), 20–29.
Bringle, R. G., & Hatcher, J. A. (1999). Reflection in service-learning: Making meaning of experience. *Educational Horizons, 77*(4), 179–185.
Brown, N. C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart: The resurgence of computer science in UK schools. *ACM Transactions on Computing Education, 14*(2), 9.
Caeli, E. N., & Yadav, A. (2019). Unplugged approaches to computational thinking: A historical perspective. *TechTrends, 64*, 29–36. https://doi.org/10.1007/s11528-019-00410-5
Codding, D., Alkhateeb, B., Mouza, C., & Pollock, L. (2021). From professional development to pedagogy: An examination of computer science teachers' culturally responsive instructional practices. *Journal of Technology & Teacher Education, 29*(4), 497–532.
Code.org, CSTA, ECEP Alliance. (2021). *2021 State of computer science education: Accelerating action through advocacy*. Retrieved from https://advocacy .code.org/stateofcs.
Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Lawrence Erlbaum Associates, Inc.
College Board. (2017). *AP computer science principles*. New York, NY: College Board.
Computer Science Teachers Association. (2017). https://www.csteachers.org/page/standards.
Corbin, J., & Strauss, A. (2015). *Basics of qualitative research: Techniques and procedures for developing grounded theory* (4th). Sage Publishing.
Creswell, J. W. (2007). *Qualitative inquiry and research design: Choosing among five approaches*. Sage publications.
Darling-Hammond, L., Hyler, M., & Gardner, M. (2017). *Effective teacher professional development*. Learning Policy Institute.
Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM, 60*(6), 33–39. https://doi.org/10.1145/2998438
Desimone, L. M. (2009). Improving impact studies of teachers' professional development: Toward better conceptualizations and measures. *Educational Researcher, 38*(3), 181–199.
Desimone, L. M., & Garet, M. S. (2015). Best practices in teachers' professional development in the United States. *Psychology, Society, & Education, 7*(3), 252–263.
Falkner, K., Vivian, R., & Falkner, N. (2014, January). The Australian digital technologies curriculum: Challenge and opportunity. In *Proceedings of the sixteenth Australasian computing education conference* (Vol. 148, pp. 3–12). Australian Computer Society, Inc.
Falkner, K., Vivian, R., & Williams, S. A. (2018). An ecosystem approach to teacher professional development within computer science. *Computer Science Education, 28*(4), 303–344. https://doi.org/10.1080/08993408.2018.1522858
Fancsali, C., Mark, J., & DeLyser, L. A. (2020). NYC CS4All: An early look at teacher implementation in one districtwide initiative. In *2020 research on equity and sustained participation in engineering, computing, and technology (RESPECT), Portland, OR, 2020* (pp. 1–8). https://doi.org/10.1109/RESPECT49803.2020.9272418
Field, A. (2013). *Discovering statistics using IBM SPSS statistics*. London: Sage Publications.
Giannakos, M. N., Doukakis, S., Pappas, I. O., Adamopoulos, N., & Giannopoulou, P. (2015). Investigating teachers' confidence on technological pedagogical and content knowledge: An initial validation of TPACK scales in K-12 computing education context. *Journal of Computers in Education, 2*(1), 43–59.
Glaser, B. G., & Strauss, A. L. (1967). *The discovery of grounded theory: Strategies for qualitative research*. Aldine Publishing.
Goode, J., Malyn-Smith, J., Peterson, K., & Chapman, G. (2020). Features that support an equity-based professional learning community. *Computing in Science & Engineering, 22*(5), 51–59.
Guskey, T. (2013). Does it make a difference? Evaluating professional development. In *Leading professional learning: Building capacity through teacher leaders, module 6*. ASCD. Retrieved from https://pdo.ascd.org/LMSCourses/PD13OC010M/media/Leading_Prof_Learning_M6_Reading1.pdf.
Israel, M., Wherfel, Q. M., Shehab, S., Ramos, E. A., Metzger, A., & Reese, G. C. (2016). Assessing collaborative computing: Development of the collaborative-computing observation instrument (C-COI). *Computer Science Education, 26*(2–3), 208–233.
ISTE. (2016). https://www.iste.org/standards/for-students.
Jacob, S., Nguyen, H., Tofel-Grehl, C., Richardson, D., & Warscauer, M. (2018). Teaching computational thinking to English learners. *NYS TESOL Journal, 5*(2), 12–24.
Kong, S. C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices and pedagogy. *Computers & Education, 151*(July 2020), 103872.
Lareau, A. (1996). Common problems in fieldwork: A personal essay. In A. Lareau, & J. Shultz Journeys (Eds.), *Through ethnography: Realistic accounts of fieldwork* (pp. 195–236). Westview Press.
Lawless, K. A., & Pellegrino, J. W. (2007). Professional development in integrating technology into teaching and learning: Knowns, unknowns, and ways to pursue better questions and answers. *Review of Educational Research, 77*, 575–614. https://doi.org/10.3102/0034654307309921
Margolis, J., Ryoo, J., & Goode, J. (2017). Seeing myself through someone else's eyes: The value of in-classroom coaching for computer science teaching and learning. *Transactions on Computing Education, 17*(2), 1–18.
Menekse, M. (2015). Computer science teacher professional development in the United States: A review of studies published between 2004 and 2014. *Computer Science Education, 25*(4), 325–350. https://doi.org/10.1080/08993408.2015.1111645
Merriam, S. B., & Tisdell, E. J. (2016). *Qualitative research: A guide to design and implementation*. John Wiley & Sons.
Mishra, P., & Koehler, M. (2006). Technological Pedagogical Content Knowledge: A Framework for Teacher Knowledge. *Teachers College Record, 108*(6), 1017–1054.
Mouza, C., Marzocchi, A., Pan, Y., & Pollock, L. (2016). Development, implementation and outcomes of an equitable computer science after-school program: Findings from middle school students. *Journal of Research on Technology in Education, 48*(2), 84–104.

Mouza, C., Pan, Y., Yang, H., & Pollock, L. (2020). A multiyear investigation of student computational thinking, practices, and perspectives in an after school computing program. *Journal of Educational Computing Research, 58*(5), 1029–1056.

Mouza, C., Sheridan, S., Lavigne, N., & Pollock, L. (2021). Expanding an equitable pedagogy framework for teaching computer science: Reflections from the field. *Computer Science Education, Ahead-of-print*, 1–26. https://doi.org/10.1080/08993408.2021.1970435

Mouza, C., Yadav, A., & Leftwich, A. (2018). Developing computationally literate teachers: Current perspectives and future directions for teacher preparation in computing education. *Journal of Technology and Teacher Education, 26*(3), 333–352.

NGSS Lead States. (2013). *Next generation science standards: For states, by states*. National Academies Press.

Nunnally, J. C. (1978). *Psychometric theory* (2nd ed.). McGraw Hill.

Penuel, W., Fishman, B., Yamaguchi, R., & Gallagher, L. P. (2007). What makes professional development effective? Strategies that foster curriculum implementation. *American Educational Research Journal, 44*(4), 921–958.

Pollock, L., Mouza, C., Atlas, J., & Harvey, T. (2015). Field experience in teaching computer science: Course organization and reflections. In *SIGCSE'15: Proceedings of 2016 ACM SIGCSE Technical Symposium on Computer Science Education, February 2015* (pp. 374–379).

Pollock, L., Mouza, C., Czik, A., Little, A., Coffey, D., & Buttram, J. (2017). From professional development to the classroom: Findings from CS K-12 teachers. In *SIGCSE'17: Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education, March 2017* (pp. 477–482).

Qian, Y., Hambrusch, S., Yadav, A., & Gretter, S. (2018). Who needs what: Recommendations for designing effective online professional development for computer science teachers. *Journal of Research on Technology in Education, 50*(2), 164–181.

Rich, P. J., Mason, S. L., & O'Leary, J. (2021). Measuring the effect of continuous professional development on elementary teachers' self-efficacy to teach coding and computational thinking. *Computers & Education, 168*(2021), 104196.

Sadik, O., Ottenbreit-Leftwich, A., & Brush, T. A. (2020). Secondary computer science teachers' pedagogical needs. *International Journal of Computer Science Education in Schools, 4*(1), 1–21.

Sentance, S., & Humphreys, S. (2018). Understanding professional learning for Computing teachers from the perspective of situated learning. *Computer Science Education, 28*(4), 345–370. https://doi.org/10.1080/08993408.2018.1525233

Shulman, L. S. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher, 15*, 4–14.

Strauss, J. M., & Corbin, J. (1990). *Basics of qualitative research: Grounded theory procedures and techniques*. Sage.

Vegas, E., Hansen, M., & Fowler, B. (2021). *Building skills for life: How to expand and improve computer science education around the world*. The Brookings Institution.

Vivian, R., & Falkner, K. (2019). Identifying teachers' Technological Pedagogical Content Knowledge for computer science in the primary years. In *Proceedings of the 2019 ACM conference on international computing education research* (pp. 147–155).

Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education, 26*(4), 235–254. https://doi.org/10.1080/08993408.2016.1257418

Yin, R. K. (2018). *Case study research and applications: Design and methods*. Sage publications.

Zhou, N., Nguyen, H., Fischer, C., Richardson, D., & Warschauer, M. (2020). High school teachers' self-efficacy in teaching computer science. *ACM Transactions on Computing Education, 20*(3), 18. https://doi.org/10.1145/3410631. Article 23 (September2020).