SWOT Analysis of Two Different Designs of Summer Professional Development Institutes for K-8 CS Teachers

Patrick Morrow
Dept. Computer Science & Engineering
University of Nebraska-Lincoln
Lincoln, USA
pmorrow@cse.unl.edu

Leen-Kiat Soh
Dept. Computer Science & Engineering
University of Nebraska-Lincoln
Lincoln, USA
lksoh@cse.unl.edu

Gwen Nugent
Nebraska Research Center for Children, Youth, Families and Schools
University of Nebraska-Lincoln
Lincoln, USA
gnugent1@unl.edu

Wendy Smith
Center of Mathematics, Science, and Computer Education
University of Nebraska-Lincoln
Lincoln, USA
wsmith5@unl.edu

Guy Trainin
Dept. Teaching, Learning, and Teacher Education
University of Nebraska-Lincoln
Lincoln, USA
gtrainin2@unl.edu

Kent Steen
Lincoln Public Schools
Lincoln, USA
ksteen@lps.org

***Abstract*—Increasingly professional development (PD) programs have been designed and implemented for pre-service and in-service teachers to acquire CS content knowledge and CS pedagogy and instructional strategies for K-12 students. This paper reports on our adaptation, implementation and research program for K-8 CS teachers across a Midwestern state. More specifically, its PD program for K-8 CS teachers consists of a summer institute with two graduate courses and a series of Saturday workshops during the subsequent academic year. This paper focuses on the two summer courses: one on CS knowledge content including computational thinking, variables, conditionals, loops, arrays, functions, and algorithms, and one instructional strategies, student pedagogy, computer-aided education resources, and community building. We report our SWOT (Strengths, Weaknesses, Opportunities, Threats) analysis of the two summer institutes involving the two courses to identify what went well and what needed improvement. This paper also reviews best practices for summer PD.**

Keywords—professional development, computer science, teachers, SWOT analysis

I. Introduction

The need for K-12 computer science (CS) instruction has become of great importance throughout the world as more and more career paths rely heavily on CS literacy. As a result, we have increasingly seen professional development (PD) programs designed and implemented for pre-service and in-service teachers to acquire both the CS content knowledge and pedagogy and instructional strategies in teaching K-12 CS. This paper reports on our adaptation, implementation and research program for K-8 CS teachers across a Midwestern state. More specifically, its PD program for K-8 CS teachers consists of a summer institute with two graduate courses and a series of Saturday workshops during the subsequent academic year. This paper focuses on the two summer courses: one on *CS knowledge content* including computational thinking, variables, conditionals, loops, arrays, functions, and algorithms, and one on *instructional strategies* including addressing issues in student pedagogy, computer-aided education resources, and community building.

This paper starts with a review of summer PD workshops or institutes that have been reported and evaluated in the literature, including short and longer courses, use of programming language and environment, the incorporation of which subsets of CS content, and other issues pertinent to prepare teachers in confidently and effectively teach K-12 teachers. Based on this review, we will also highlight best practices for summer PD.

This paper also reports on two iterations of the summer institutes designed in alignment with best practices. The first institute was held in Summer 2019 over a two-week period, teaching two courses to a cohort of 29 teachers. The second institute was held in Summer 2020, also over a two-week period, with the same two courses, to a cohort of 24 teachers. We report our SWOT (Strengths, Weaknesses, Opportunities, Threats) analysis, a proven analysis tool [13] of the two summer institutes to identify strengths and challenges to implementation. The strengths focus on the successes. The weaknesses pinpoint areas where that need to improve. The opportunities focus on possible improvements based on feedback, insights, and experiences. The threats highlight potential challenges to the success of the program. We used the SWOT analysis of the first summer institute in revising the program design of the second summer institute.

Moreover, due to the disruptions caused by the Covid-19 pandemic, the second summer institute was conducted entirely online. The PD had to be re-designed to meet relevant constraints such as the lack of in-person team building, the absence of physical instructional resources–instructional robots, and at-home distractions. Along with our evaluation data, our SWOT analysis of the second summer institute thus also includes recommendations for effective online PD.

II. Related Work and Background

The National Science Foundation has been targeting CS participation in K-12 through the CS for All [8,25,27] and CS10K [5,29] initiatives. Qualified CS teachers are vital to creating an infrastructure for integrating CS into K-12. The shortage of qualified teachers at all levels (e.g., per reports by Code.org) has led to efforts to develop PD programs that effectively prepare current teachers to teach CS. Teachers are still going into their classrooms under-prepared to teach CS. Ericson et al. found such deficiencies in two of their CS PD workshops [7]. The first workshop was for teachers with little or no CS teaching experience, and the second was for teachers of a CS-AP high school course. After the first course, 70.37% of teachers felt more capable in programming, 96.03% had a better idea of what to teach, and 88.89% got a better idea of how to teach CS. However, only 44.44% of the teachers felt ready to teach CS. Of the 17 teachers from the CS-AP workshop, 94.12% reported feeling more capable in programming, 88.24% has a better idea of what to teach, and 94.12% had a better idea of how to teach CS. 76.47% of the CS-AP teachers felt ready to teach CS in the next school year. Overall, in their summer PD workshop for CS teachers, they found, post-workshop, that 56.82% of the teachers felt ready to teach CS in the next semester [7]. Even with an increase in programming and pedagogy knowledge, many teachers are still preparing to teach students with little confidence in their ability to do so. Ericson et al. also found that 29% of all teachers wanted the workshop to go at a slower pace. Going forward, they suggested that creating a program that caters to the new introductory CS teachers needs a slower pace to improve PD outcomes [7].

Research has identified ways to increase self-efficacy and use of computers in classrooms. Hatlevik et al. found there was a strong positive correlation between the amount of home computer use and ICT self-efficacy, which is vital to learning CS and learning to teach CS [12]. Wozney et al. also saw teachers with personal computers and access to "play with" potential classroom tools were more likely to integrate technology in the classroom [28]. However, most PD programs (e.g., [1,21]) do not explore the impact of differences between teachers with experience teaching CS (or experience using CS tools to teach other subjects) and teachers without CS education backgrounds. The study detailed in this paper makes such comparisons to provide insight into the relationship between teacher CS experience and their CS knowledge, attitudes, and skills.

A valuable PD approach is the Exploring Computer Science (ECS) PD program used by McGee et al. [18]. The ECS curriculum was designed for teachers to teach high school students CS through equity, inquiry, and CS concepts. The curriculum introduces CS through real-world examples, such as making games that encourage learning about healthy eating [18]. The PD program's workshop had five key components. The first two components focus on active learning [6], the third focuses on equity in CS education, and the last two focus on making the teachers successful in the long term. McGee et al. used an Expectancy-Value-Cost (EVC) survey to measure the attitudes of the high school ECS students. They compared the EVC survey results to the students' course experience—in the courses taught by the participating teachers—and to a Teaching Quality Index (TQI) based on a combination of two teacher practice quality instruments to measure the teachers' ability to "foster equity, inquiry, and development of CS concepts" [18]. The students took the survey to determine the teachers' TQI scores. The authors found the TQI had a direct effect on the students' post-EVC scores, which in turn influences student outcomes. This finding shows that better-equipped teachers are having a direct impact on students' attitudes and their engagement in CS. Additionally, the more experience the teachers had in teaching ECS, the more the students' ECS scores improved from the pretest [18]. McGee et al.'s method of measuring teacher performance and student learning outcomes could help in creating a universal measure for K-12 CS educators. In our project, we also adopt this approach to measure teacher performance that also includes impact on student learning and performance.

A. CS Content of CS PD Programs

One of the goals of CS PD programs is to balance depth and breadth of CS knowledge in preparing pre- and in-service CS teachers. Program designers use two general approaches to achieve this goal. The first approach is through *programming language training*, where the teachers learn CS concepts through programming in high-level CS languages. The second approach is through *CS unplugged activities*. These activities can include CS concepts but focus on *computational thinking* (CT) to introduce teachers to CS. CT draws on skills and professional practices that are fundamental to computer science without focusing on the specific syntax and practices of computer language that can become an obstacle [26]. The CS unplugged approach allows teachers from all CS backgrounds to understand CS concepts without needing to learn a programming language or use any devices [2]. Furthermore, the extent to which CS or C content are covered also depends on the duration of PD programs.

Short PD programs are one week or less. This is often done out of practical concerns about teacher time (e.g. teacher summer schedules) and funding. Some programs are as short as 1-3 days [3,21]. During such a short time, there is not enough time to cover substantial CS concepts. The 1-3-day programs have been successful by focusing on training teachers to use high quality curriculum and tools that they can apply to their classrooms right away. This type of program makes sense to improve the preparedness of teachers already equipped with adequate CS backgrounds. For example, Morreale et al.'s two 1-day workshops helped introduce teachers to CT by providing them sessions on curriculum materials, current university projects, internships, post-grad opportunities, and the importance of CS locally and nationally [21]. Bower et al. held four separate 1-day workshops for teachers of grades K-2, 3-4, 5-6, and 7-8 [3]. The teachers were taught CT concepts, and strategies and technologies used to teach them. These two short PD programs reported significant self-efficacy improvements made in a short amount of time. While this improvement is encouraging, given the growing need for CS teachers, we argue that introducing teachers to the CT concepts over a 1-3-day workshop might not be enough to prepare teachers for providing quality CS instruction.

Medium length PD programs should be able to expand on the successes of the short PD programs by going more in-depth. Liu et al. used a 5-day game-centered development approach and a drag-and-drop programming language called Stencyl to prepare their teachers [17]. Each of the five days contained two sessions, and each session contained one or two CS concepts. The concepts covered were classes, variables, methods, conditionals, Booleans, loops, and lists. In the mornings, the teachers worked on existing Stencyl projects that covered the concept of the day. In the afternoons, the teachers created their curriculum for the concept using Stencyl to take back to their classrooms. Liu et al.'s team reported a 61% increase in concept knowledge [17]. In another example, Pollock et al. designed their 4.5-day PD program with a focus on CS content, pedagogical strategies for teaching CS, and strategies for broadening participation in CS [24]. Both programs reported increases in knowledge, although the two programs had slightly different goals. Pollock et al. focused on connecting CS and CS pedagogy while Liu et al. focused on content knowledge and mastery of a programming language (namely, Stencyl). One interesting thing to note in the medium-length programs is that *the extended length of the program allows for more creativity in the program design*.

With more time and added program flexibility, *long PD programs* allow for added depth and breadth of knowledge. There was an increase in variety in the design of PD programs as the programs went from short to medium, so the long PD programs are expected to introduce even more range in goals, instructional strategies, and workshop tools. Milliken et al. found success with their reworked two-week PD program. The program focused less on purely CS content, and more on a *Lead Learner* model where one group of teachers acts as the instructors, and the other groups act as the learners. The *Lead Learner* model helps all teachers participate as both instructors and students throughout the program [20]. Goode et al. found success using the ECS model for PD and curriculum design in their two-year PD program. In the first year, the authors held a one-week PD program with quarterly follow-up sessions post-program. In year two, the authors held a second one-week program [9]. Scratch, Lego Mindstorms, and CS Unplugged activities are typically used in ECS classes to deliver concepts of CS without having to spend much time learning a programming language, [9]. The ECS model also strives to form long-term relationships with teachers. These two PD programs achieved high-levels of teacher preparedness by teaching about CS concepts and linking them to the classroom and the pedagogy that teaches the teachers how to deliver a specific curriculum. With the added length of the program, the designers can follow a specific curriculum that helps the teachers understand what they will need to teach in their classroom and how they will need to teach it.

When designing PD, it is necessary first to identify the goals and identify any limitations. Examples of limitations could be duration, participant prior CS knowledge, and school system restrictions. After reviewing the limitations, PD leaders can design the program structure including the PD length, intensity, sequence of topics, and assessments. For programs of all lengths, it is necessary to provide support for the teachers throughout their journey of implementing CS in their classrooms. The initial PD preparation can only take the teachers so far, and questions will inevitably arise as the teachers begin implementing the learned materials in their classrooms. Bower et al. found that participants indicated the need for "peer mentoring networks," and Pollock et al.'s participants expressed a need for collaboration and communication amongst peers [3, 24]. The long-term projects by Milliken et al. and Goode et al. had long-term facilitator/participant relationship embedded as part of their program [9, 20]. A support-network post-program is a theme throughout successful professional development programs.

B. *Types of Programming of CS PD Programs*

CS PD programs teaches programming to teachers to use and implement CS concepts and expose them to the process of developing programs to solve problems. Several programs incorporate programming languages such as Python, JavaScript, Java, or other high-level languages to introduce CS concepts. In contrast, others use more CS-unplugged (no technology needed) approaches paired with visual programming languages such as Blockly, Scratch. The programs reviewed in this section will uncover the differences between using text-based programming languages vs. visual programming languages to teach CS concepts to K-12 teachers.

PD programs that used visual programming language include the ECS curriculum [9,18,19] and block-based programming languages [4,14]. In these visual programming language-centered programs, we observe that there was *more emphasis on CT concepts*, compared to PD programs that used high-level programming languages. Noone and Mooney (2018) noted that researchers tend to agree that visual programming languages fall short when facing complex CS [23]. While this may be true, visual programming languages have been a successful tool when introducing teachers to CT concepts, as verified by Brennan and Resnick [4].

Text-based programming languages encourage a deeper understanding of CS concepts to solve many problems compared to visual-based programming languages. Lee et al. held a year-long PD program for 66 in-service high school STEM teachers [15]. The goal of the program was to teach content and scientific practices in the spring and pedagogy and recruitment techniques during the summer. The outcomes from this program show that the program did an excellent job of engaging the teachers in CS practice and exposing teachers to new ways of adopting CS. Desmoine and Garet have found that PD is more successful when it is explicitly linked to classroom lessons [6]. This link can be challenging to make when facilitating a PD program using a text-based programming language especially if the programming language is not an instructional tool used by the teachers in their classrooms. Another program that was heavily content-focused using text-based programming languages was designed by Leyzberg and Moretti [16]. Their goal was to offer a content-focused PD opportunity for teachers that lack strong CS backgrounds. The program was adapted from a college CS course to cover a week worth of content each day. The lectures provided hands-on experience with CS concepts, practice applying the concepts, and first steps towards creating assignments. The concepts taught during the PD were advanced: input/output, recursion, algorithm, data structure analysis, keyvalue data structures, Boolean logic, decimal/hexadecimal/binary conversions, machine learning, intractability (P vs. NP and NP-completeness), and circuit design [16]. This program was fast-paced and covered some advanced CS concepts. Overall, *in the text-based programming language programs, we see more difficult concepts being covered during the programs*. Additionally, these programs are typically longer (one week or longer). Any shorter than one week, and the teachers likely will not have time to gain deep understanding of concepts and the programming language. Both PD programs [15, 16] were found to be beneficial to the participants and well-received.

In both text-based and visual programming language programs, researchers report significant increases in content knowledge scores [e.g., 15, 16, 18, 19]. Both program types also saw similar positive feedback about the program design. In terms of a content knowledge advantage, it is difficult to find one between the two program types because each program uses a different measure. There seem to be two determinants for using one design over the other. This first is the allotted program length; any program under one weeklong will have a harder time introducing a text-based programming language. The other determinant is the goal of the participants and the program designers. Grades 6-8 teachers may require text-based programming experience to effectively teach their classrooms, whereas grade K-5 teachers may only need visual programming experience.

III. The Two Summer PD Institutes

In this paper, we review two summer professional development institutes, one for each cohort: summer 2019 and summer 2020. Each summer institute lasted for 2 weeks, thus making each a long PD program as outlined in Section II. The Cohort 1 PD design decisions were experimental relying on best practices known at the time and the expectation that the results of the first cohort will help guide the cohort 2 redesign. Thus, we anticipated to make changes in our PD design from the Cohort 1 PD program to the Cohort 2 PD program. Furthermore, we planned to make only a small number of changes from Cohort 1 to Cohort 2, so comparisons could be evaluated. However, due to the COVID-19 pandemic, our design changed drastically. Arguably the most significant change from Cohort 1 to Cohort 2 was the change from in-person to online instruction.

One of the key non-pandemic-related changes was in sequencing. In the first summer institute, the CS content course was held during the first week and the CS pedagogy course was held in the second week. For the second summer institute, both courses were held for half-days over two weeks: The CS content course was held in the morning, and the CS pedagogy course was held in the afternoon.

A second change was in the lead instructors. In the first summer institute, the lead instructor was a university CS professor with four teaching assistants who were CS university students. In the second summer institute, we replaced the lead instructor with a high school teacher from Cohort 1's CS pedagogy instruction team with two top-performing teachers from Cohort 1 and 3 student teaching assistants.

Finally, in the first summer institute, we taught Python using the PyCharm Integrated Development Environment (IDE). We chose Python because the syntax is simple and is widely discussed as a first programming language for beginners. However, our teachers had challenges with PyCharm and Python versions on the teachers' local computers due to installation differences. Additionally, at the beginning of the program, the lead instructor and the teachers were using two separate IDEs for Python development. This caused confusion during lectures and coding examples. This issue was mitigated early in the CS content course as the lead instructor switched to using the same IDE when demonstrating and testing code. In the second summer institute, the new lead instructor chose to change the language to JavaScript and use the internet tool, JSFiddle, as an IDE. The new language and IDE worked great for several reasons. First, JSFiddle is widely available, and once a free account is created, all the work is saved on the site. JSFiddle did not require any set-up instructions nor installation, which made the introduction to coding near-seamless. Quickly onboarding the teachers with JSFiddle was a crucial step to engage teachers right away, especially those coming into the PD program with high apprehension. Lastly, JavaScript, like Python, is regarded as another good programming language for beginners.

All teachers in our summer institutes taught at the K-8 grade levels, with a significant majority in the elementary levels. Table I shows the demographic information about Cohort 1 and Cohort 2 teachers.

TABLE I. Demographic Information about Cohort 1 and Cohort 2 Teachers

	Gender		Grade Level of Instruction		District	
Cohort	M	F	K-5	6-12	Urban	Rural
1 (N=28)	7	21	14*	14*	18	10
2 (N=24)	4	20	19*	12*	0	24

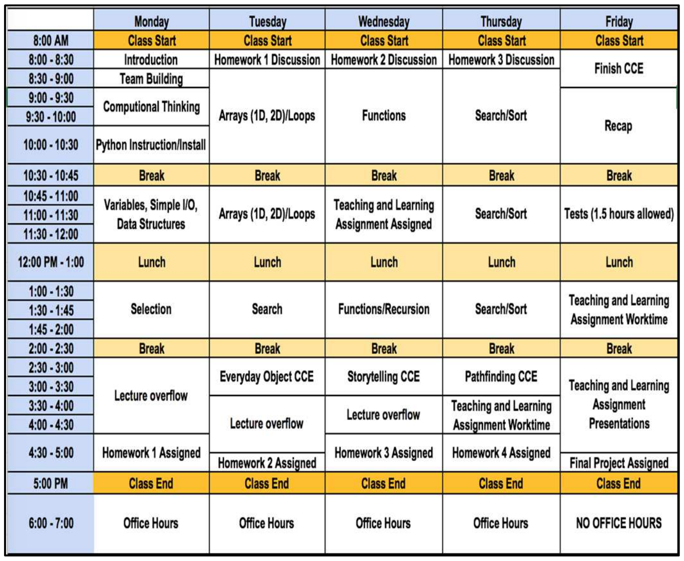
* There were teachers who taught in both K-5 and 6-12 grade levels.

IV. SWOT Analysis of the First Summer Institute

This section reports on the SWOT (Strengths, Weaknesses, Opportunities, Threats) analysis of the first summer institute, a proven analysis tool [13], used to identify what went well and what needed improvement. The strengths section of SWOT focuses on the successes. The weaknesses section pinpoints areas that need to improve. The opportunities section focuses on possible improvements based on feedback, insights, and experiences. The threats section highlights potential threats to the success of the program. SWOT analysis was used to help inform decisions made about the next PD program delivery. Figures I-II show a summary of the topics and activities covered in the first summer institute.

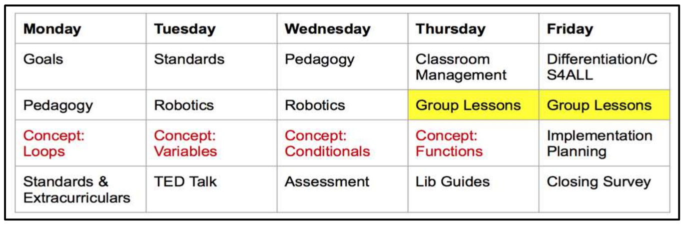
A. Strengths

The instructional teams were large enough to support learning expectations. Instructional teams included one faculty instructor, one graduate TA, and three undergraduate TAs for the first-week course (CS), and four master teachers as instructors for the week-2 course (CS pedagogy) to help all teachers promptly. The instructional team was adaptive to the teachers' needs throughout the two courses. They created new examples and altered course content using a just-in-time teaching approach [22] to fit learners' needs.



	Monday	Tuesday	Wednesday	Thursday	Friday
8:00 AM	Class Start	Class Start	Class Start	Class Start	Class Start
8:00 - 8:30	Introduction	Homework 1 Discussion	Homework 2 Discussion	Homework 3 Discussion	Finish CCE
8:30 - 9:00	Team Building	Arrays (1D, 2D)/Loops	Functions	Search/Sort	
9:00 - 9:30	Computional Thinking				Recap
9:30 - 10:00					
10:00 - 10:30	Python Instruction/Install				
10:30 - 10:45	Break	Break	Break	Break	Break
10:45 - 11:00	Variables, Simple I/O, Data Structures	Arrays (1D, 2D)/Loops	Teaching and Learning Assignment Assigned	Search/Sort	Tests (1.5 hours allowed)
11:00 - 11:30					
11:30 - 12:00					
12:00 PM - 1:00	Lunch	Lunch	Lunch	Lunch	Lunch
1:00 - 1:30	Selection	Search	Functions/Recursion	Search/Sort	Teaching and Learning Assignment Worktime
1:30 - 1:45					
1:45 - 2:00					
2:00 - 2:30	Break	Break	Break	Break	Break
2:30 - 3:00	Lecture overflow	Everyday Object CCE	Storytelling CCE	Pathfinding CCE	Teaching and Learning Assignment Presentations
3:00 - 3:30					
3:30 - 4:00		Lecture overflow	Lecture overflow	Teaching and Learning Assignment Worktime	
4:00 - 4:30					
4:30 - 5:00	Homework 1 Assigned	Homework 2 Assigned	Homework 3 Assigned	Homework 4 Assigned	Final Project Assigned
5:00 PM	Class End	Class End	Class End	Class End	Class End
6:00 - 7:00	Office Hours	Office Hours	Office Hours	Office Hours	NO OFFICE HOURS

FIGURE I. SCHEDULE OF WEEK 1 OF THE FIRST SUMMER INSTITUTE: FOCUSING ON CS AND CT TOPICS AND ACTIVITIES



Monday	Tuesday	Wednesday	Thursday	Friday
Goals	Standards	Pedagogy	Classroom Management	Differentiation/CS4ALL
Pedagogy	Robotics	Robotics	Group Lessons	Group Lessons
Concept: Loops	Concept: Variables	Concept: Conditionals	Concept: Functions	Implementation Planning
Standards & Extracurriculars	TED Talk	Assessment	Lib Guides	Closing Survey

FIGURE II. SCHEDULE OF WEEK 2 OF THE FIRST SUMMER INSTITUTE: FOCUSING ON PEDOGIGAL TOPICS AND ACTIVITIES. NOTE THAT THE SCHEDULING OF WEEK 2 OF THE FIRST SUMMER INSTITUTE WAS LESS PRESCRIBED AND DETAILED COMPARED TO OTHER SCHEDULES PRESENTED IN THIS PAPER.

The 28 teachers from a local school district took the same pre- and post-program test over CT and CS concepts to measure their knowledge gained. The teachers who took the test had CS experience before the course. An earlier report showed that the summer CS PD program had a positive impact on the teachers' CT and CS concept knowledge [10].

The program continued during the academic year and into the following summer, which gave the teachers more resources and time to learn the CT and CS concepts. A Virtual Community was set up through Listserv so the teachers could collaborate, share ideas, and ask each other for help after the course ended. During the subsequent academic year, the teachers met five times to reinforce the PD learning. The teachers reviewed CT and CS concepts learned over the summer, shared class materials, and connected with other teachers.

Observations and feedback from the professional development showed that teachers were able to help each other understand difficult concepts. K-8 teachers became experts at breaking down difficult concepts into terms that were understood by their peers.

B. *Weaknesses*

The first-week course used lecture-based learning mixed with hands-on group activities and programming tasks, but the lecture aspect did not engage the teachers. Teachers learned best when short, brief lectures were followed by learning activities. Using the just in time teaching we incorporated more learning activities than initially planned.

The goals of the instructor and the goals of the teachers did not align during this course. The instructor hoped the teachers would become capable programmers while learning CS and CT concepts. The teachers hoped to learn how to teach CS concepts to their students. The teachers were not prepared to learn the concepts through programming and had a difficult time with the programming language. Syntax and abstraction aspects made the language an impractical approach for engaging K-8 teachers learning about CS and CT concepts. Teachers missed the opportunity to link the concepts learned each day to future CS classroom instruction since the CS concepts and the CS pedagogy were taught separately.

We created a Slack channel, a Cloud-based instant messaging software, as a virtual community after the program, but the teachers did not make use of the site. We hypothesized that the lack of engagement was due to the teachers' unfamiliarity with Slack. As a result, the virtual community was moved to Listserv, a more accessible service that connects groups of people through their email. Both attempts to create a virtual learning community have fostered little to no communication. Since active virtual learning community is an important component that needs to be developed for future PD programs.

The first-week CS content course covered basic concepts like strings, variables, conditions, and loops before progressing to more complicated concepts like functions, recursion, sorting, and searching. After covering the basic concepts, the teachers still had difficulty with loops and conditionals. Therefore, the teachers were not prepared for the transition to the more difficult concepts.

C. *Opportunities*

Data collection tools could be restructured for the next cohort for smoother data analysis to support in-class intervention. For example, services such as Google Forms can be used to collect teacher responses for their assignments and store them all in one place and the same format so that the data analysis process would be efficient and provide more efficient in-time feedback during the summer institutes.

All elementary and middle school teachers received funding to purchase CS instructional hardware and software as part of participating in the PD program. The first cohort used the available funds to purchase educational robots and tablets. The multiple feedback opportunities would show how new educational tools had been utilized and how the feedback could then be used in future programs to better familiarize others teachers with tools they could be using.

D. *Threats*

Over the two weeks, material covered needed to be reduced to accommodate the speed the teachers were learning. Thus, the material may have been altered to the point that not all the CS concepts specified in the course requirement were taught in-depth or at the intended level of rigor, though all basic CS concepts were covered. For example, at the beginning of the course, basic concepts (variables, Boolean logic, conditionals, loops, functions) and some advanced concepts (recursion, file I/O) were planned, but after just in time feedback the advanced concepts, recursion, and file I/O, were only briefly covered.

The teachers collaborated effectively on most assignments, preferred to continue to work together on their assignments. The teachers' collaboration made it challenging to design and facilitate individual work and collect comprehensive individual measures of CS and CT knowledge (e.g., assignments on reflection, analysis, and programming) in addition to the individual end-of-course knowledge tests.

The teachers had varying levels of experience with CS and taught different grade levels. Catering materials to each grade level and experience level was a challenge.

V. SWOT Analysis of the Second Summer Institute

This section reports on the SWOT analysis on the second summer institute. Note that as discussed in Section III, the second summer institute was carried out via a virtual platform (Zoom) due to the pandemic. Figures III-VI show a summary of the topics and activities covered in the second summer institute. Similarly, an earlier report showed that the PD program had a positive impact on the teachers' CT and CS concept knowledge [11].

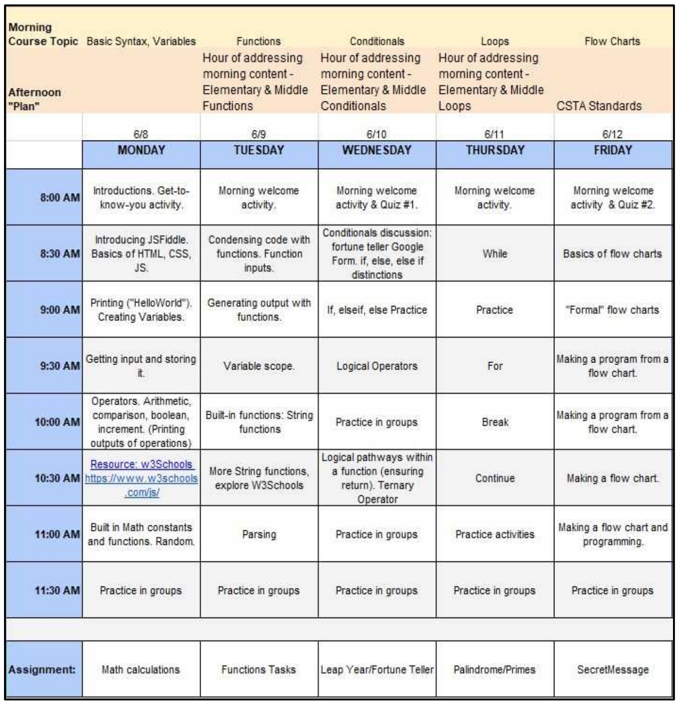
A. Strengths

JavaScript and JSFiddle.com made programming more approachable as opposed to Python and the IDE used for the first cohort. There was minimal setup to begin coding and made it easier for teachers to start coding. Using JavaScript allowed many of the Cohort 2 participants to feel comfortable programming in just two weeks.

The facilitators of the program used breakout rooms through Zoom to allow the teachers to work in groups on daily activities. The breakout rooms always had at least one facilitator and no more than five teachers to a room. These breakout rooms helped alleviate the awkwardness of video instruction and yielded valuable discussions and collaboration throughout the course. These breakouts also broke up the lectures where teachers could practice hands-on learning and reinforce each lecture topic promptly.

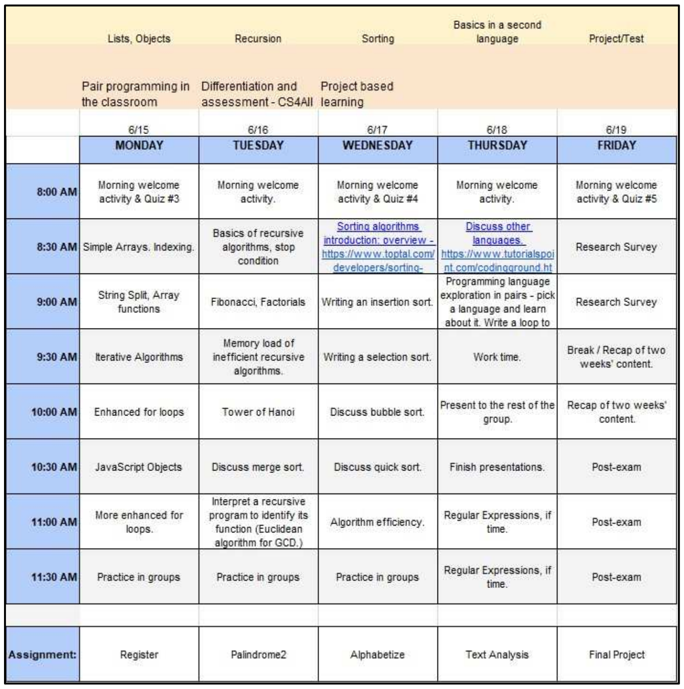
Another unforeseen benefit of online instruction was the ease of collaboration through screen sharing. Problem-solving through observation of other's code helped each teacher to understand better where their problem areas. In a traditional classroom, the facilitators would go to each teacher's desk and look at their code with them. With the online instructional format, all discussion participants can view the screen at the same time without having to move seats or leave their work.

The 2-week of half-day schedule was adequate for the facilitators to cover all CS concepts and CS pedagogy without rushing through any of the concepts. The program duration also allowed for the concepts to be linked with the pedagogy in the afternoon, which allowed the teachers to think about how they might apply the concepts they just learned into their classrooms. The duration also allowed for more robust programming assignments to be administered since the teachers were well-acquainted with each concept during the day.



Morning Course Topic	Basic Syntax, Variables	Functions	Conditionals	Loops	Flow Charts
Afternoon "Plan"		Hour of addressing morning content - Elementary & Middle Functions	Hour of addressing morning content - Elementary & Middle Conditionals	Hour of addressing morning content - Elementary & Middle Loops	CSTA Standards
	6/8	6/9	6/10	6/11	6/12
	MONDAY	**TUESDAY**	**WEDNESDAY**	**THURSDAY**	**FRIDAY**
8:00 AM	Introductions. Get-to-know-you activity.	Morning welcome activity.	Morning welcome activity & Quiz #1.	Morning welcome activity.	Morning welcome activity & Quiz #2.
8:30 AM	Introducing JSFiddle. Basics of HTML, CSS, JS.	Condensing code with functions. Function inputs.	Conditionals discussion: fortune teller Google Form. if, else, else if distinctions	While	Basics of flow charts
9:00 AM	Printing ("HelloWorld"). Creating Variables.	Generating output with functions.	If, elseif, else Practice	Practice	"Formal" flow charts
9:30 AM	Getting input and storing it.	Variable scope.	Logical Operators	For	Making a program from a flow chart.
10:00 AM	Operators. Arithmetic, comparison, boolean, increment. (Printing outputs of operations)	Built-in functions: String functions	Practice in groups	Break	Making a program from a flow chart.
10:30 AM	Resource: w3Schools https://www.w3schools.com/js/	More String functions, explore W3Schools	Logical pathways within a function (ensuring return). Ternary Operator	Continue	Making a flow chart.
11:00 AM	Built in Math constants and functions. Random.	Parsing	Practice in groups	Practice activities	Making a flow chart and programming.
11:30 AM	Practice in groups	Practice in groups	Practice in groups	Practice in groups	Practice in groups
Assignment:	Math calculations	Functions Tasks	Leap Year/Fortune Teller	Palindrome/Primes	SecretMessage

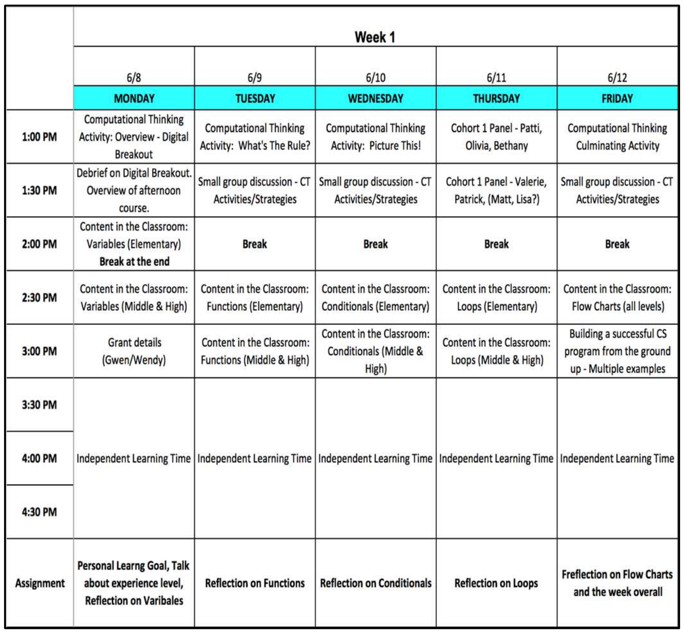
FIGURE III. MORNING SCHEDULE OF WEEK 1 OF THE SECOND SUMMER INSTITUTE: FOCUSING ON CS AND CT TOPICS AND ACTIVITIES



	Lists, Objects	Recursion	Sorting	Basics in a second language	Project/Test
	Pair programming in the classroom	Differentiation and assessment - CS4All	Project based learning		
	6/15	6/16	6/17	6/18	6/19
	MONDAY	**TUESDAY**	**WEDNESDAY**	**THURSDAY**	**FRIDAY**
8:00 AM	Morning welcome activity & Quiz #3	Morning welcome activity.	Morning welcome activity & Quiz #4	Morning welcome activity.	Morning welcome activity & Quiz #5
8:30 AM	Simple Arrays. Indexing.	Basics of recursive algorithms, stop condition	Sorting algorithms introduction: overview - https://www.toptal.com/developers/sorting-	Discuss other languages. https://www.tutorialspoint.com/codingground.ht	Research Survey
9:00 AM	String Split, Array functions	Fibonacci, Factorials	Writing an insertion sort.	Programming language exploration in pairs - pick a language and learn about it. Write a loop to	Research Survey
9:30 AM	Iterative Algorithms	Memory load of inefficient recursive algorithms.	Writing a selection sort.	Work time.	Break / Recap of two weeks' content.
10:00 AM	Enhanced for loops	Tower of Hanoi	Discuss bubble sort.	Present to the rest of the group.	Recap of two weeks' content.
10:30 AM	JavaScript Objects	Discuss merge sort.	Discuss quick sort.	Finish presentations.	Post-exam
11:00 AM	More enhanced for loops.	Interpret a recursive program to identify its function (Euclidean algorithm for GCD.)	Algorithm efficiency.	Regular Expressions, if time.	Post-exam
11:30 AM	Practice in groups	Practice in groups	Practice in groups	Regular Expressions, if time.	Post-exam
Assignment:	Register	Palindrome2	Alphabetize	Text Analysis	Final Project

FIGURE V. MORNING SCHEDULE OF WEEK 2 OF THE SECOND SUMMER INSTITUTE: FOCUSING ON CS AND CT TOPICS AND ACTIVITIES

Coupling the two courses each day, programming was learned in the morning and could be reinforced in the afternoon of each day as a practice in computational thinking: algorithmic (being methodical, creating a flowchart), problem decomposition (functions, creating a flowchart), evaluation (debugging, analysis of correctness), pattern recognition (connecting the dots, leveraging what has been learned syntax-wise, assimilating similar bugs), generalization (seeing similar problems in syntax errors, learning useful debugging approaches), and abstraction (the use of variables, the use of arrays to store values, the use of functions, the representation of mathematical equations using variables). Coupling the courses together also helped motivate teachers to appreciate and recognize the need to learn how to program to teach with more confidence and readiness, even when they are only teaching CS to grades K-5 and especially for teachers teaching CS to grades 6-8.



	Week 1				
	6/8	6/9	6/10	6/11	6/12
	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
1:00 PM	Computational Thinking Activity: Overview - Digital Breakout	Computational Thinking Activity: What's The Rule?	Computational Thinking Activity: Picture This!	Cohort 1 Panel - Patti, Olivia, Bethany	Computational Thinking Culminating Activity
1:30 PM	Debrief on Digital Breakout. Overview of afternoon course.	Small group discussion - CT Activities/Strategies	Small group discussion - CT Activities/Strategies	Cohort 1 Panel - Valerie, Patrick, (Matt, Lisa?)	Small group discussion - CT Activities/Strategies
2:00 PM	Content in the Classroom: Variables (Elementary) **Break at the end**	**Break**	**Break**	**Break**	**Break**
2:30 PM	Content in the Classroom: Variables (Middle & High)	Content in the Classroom: Functions (Elementary)	Content in the Classroom: Conditionals (Elementary)	Content in the Classroom: Loops (Elementary)	Content in the Classroom: Flow Charts (all levels)
3:00 PM	Grant details (Gwen/Wendy)	Content in the Classroom: Functions (Middle & High)	Content in the Classroom: Conditionals (Middle & High)	Content in the Classroom: Loops (Middle & High)	Building a successful CS program from the ground up - Multiple examples
3:30 PM	Independent Learning Time	Independent Learning Time	Independent Learning Time	Independent Learning Time	Independent Learning Time
4:00 PM					
4:30 PM					
Assignment	**Personal Learng Goal, Talk about experience level, Reflection on Varibales**	**Reflection on Functions**	**Reflection on Conditionals**	**Reflection on Loops**	**Frefletion on Flow Charts and the week overall**

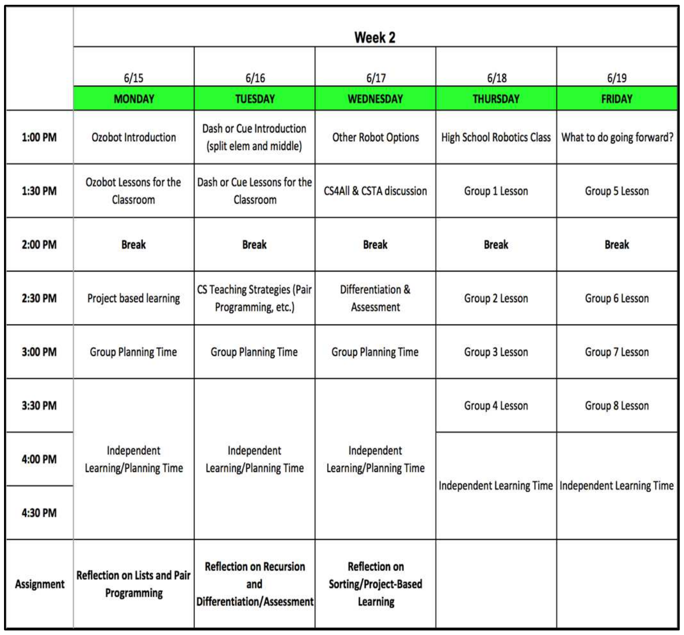
Figure IV. Afternoon schedule of Week 1 of the Second Summer Institute: Focusing on pedagogy topics, instructional resources, and activities

B. Weaknesses

Explaining more intricate concepts was made increasingly difficult, without the ability to draw on a whiteboard. Many times, a visual representation of a concept is easier to understand, and providing that was made more difficult through online instruction. The facilitators were forced to find new ways to explain concepts in detail. Though Zoom provided annotations on-screen, it was not easy or efficient to draw using a touchpad.

During the breakout rooms, there were times when one of the facilitators was unable to answer a teacher's question. In a traditional classroom, the facilitator might call over another facilitator to try to explain the answer in a different way to assist the teacher. With the breakout rooms, that facilitator-to-facilitator interaction did not occur. The facilitators instead used a back channel (on Slack) to interact.

No virtual community was established for the teachers to share ideas post-program and collaborate as they started creating lesson plans for the upcoming school years. We suspect that it might due to the nature of the online summer institute preventing the teachers from bonding as closely as they would in-person. We expected that some of the teachers exchanged emails or phone numbers, but we also expected that some teachers did not and would therefore need to communicate with the facilitators for help throughout the year.



	Week 2				
	6/15	6/16	6/17	6/18	6/19
	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
1:00 PM	Ozobot Introduction	Dash or Cue Introduction (split elem and middle)	Other Robot Options	High School Robotics Class	What to do going forward?
1:30 PM	Ozobot Lessons for the Classroom	Dash or Cue Lessons for the Classroom	CS4All & CSTA discussion	Group 1 Lesson	Group 5 Lesson
2:00 PM	**Break**	**Break**	**Break**	**Break**	**Break**
2:30 PM	Project based learning	CS Teaching Strategies (Pair Programming, etc.)	Differentiation & Assessment	Group 2 Lesson	Group 6 Lesson
3:00 PM	Group Planning Time	Group Planning Time	Group Planning Time	Group 3 Lesson	Group 7 Lesson
3:30 PM	Independent Learning/Planning Time	Independent Learning/Planning Time	Independent Learning/Planning Time	Group 4 Lesson	Group 8 Lesson
4:00 PM				Independent Learning Time	Independent Learning Time
4:30 PM					
Assignment	**Reflection on Lists and Pair Programming**	**Reflection on Recursion and Differentiation/Assessment**	**Reflection on Sorting/Project-Based Learning**		

Figure VI. Morning schedule of Week 1 of the Second Summer Institute: Focusing on pedagogy topics, instructional resources, and activities

Many of the teachers expressed confusion as to the goal of the PD program, which might have been exacerbated by the online summer institute as teachers did not have a chance to meet in-person with the instructor or the instruction team to clarify their confusion. The initial confusion was the expectation that the teachers would learn to program in addition to learning about CS concepts. Elementary teachers especially were surprised by this since they would not likely be teaching their students to program. The expectations must be made clear right away, so the teachers come into the program with the right mindset to approach the challenge of learning CS and programming concepts. This is especially important if the summer institute is held online.

C. Opportunities

Many of the participants in this cohort did not have experience teaching CS and thus did not have lesson plans specific to their classes to which they could refer as an anchor for them to incorporate changes. It was intriguing to see how they adapted what they learned in the program to their classrooms. Throughout the subsequent academic year, there were opportunities for the teachers to share their successes and failures, especially considering the disruptions caused by the pandemic. This opportunity will provide insight into the teachers' process of creating curriculum material from the PD program instruction and the validity of teaching CS through online tools, like Zoom and Canvas.

D. Threats

Since the program was delivered online, there is no way to know if the teachers used outside sources to aid them during the individual assessments at the end of the program. Measures were taken to combat collaboration between teachers during the assessments (muting all teachers and disabling chat features). Though the team did not observe any such activities during the individual assessments, this would remain as a validity concern going forward.

Due to the pandemic, the summer institute was moved to online delivery. The online instruction was a significant change to the format of the program and made it difficult to compare the outcomes of the two programs since the presentation formats are vastly different.

One challenge was that with the online format many of the teachers participated in the program from their own homes. With the ability to turn off the video, teachers might have been stepping away during lectures to deal with family issues. We have no way of knowing the amount of time the teachers were away from the screen during the lecture, and what content that they might have missed as a result.

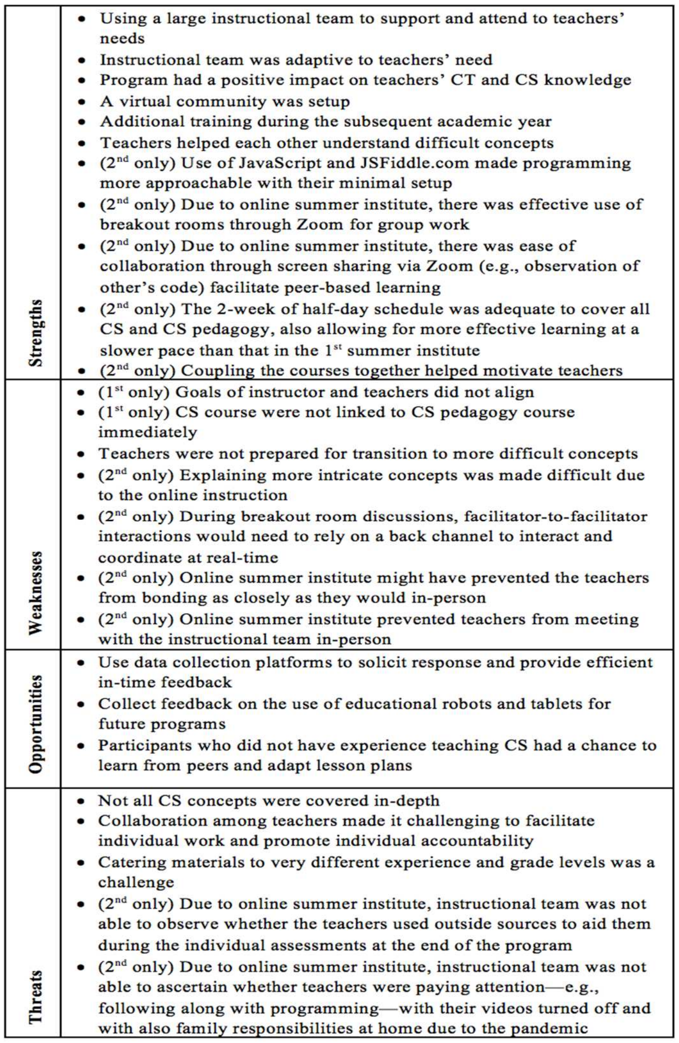
Another challenge was that only one person in a small group could talk at any time in an online platform. Teachers who were more willing to let others talk would stay silent for long periods. The ability to mute the camera and microphone in Zoom made it challenging to gauge teachers' level of engagement within each group. (Note that in an in-person classroom, it would be easier for instructor team members to observe whether members of a team participated or whether pairs of members of the same team engaged in conversations.) While the groups were sharing code, the facilitators assumed that all teachers were following along. To check each teacher's code during the small group session would have taken too much time, but some teachers might not have followed along with the code. Therefore, it would have been easy for a teacher to skip practice sessions, which would yield lower confidence and knowledge scores.

VI. Discussion

In many ways, our SWOT analysis of the first summer institute helped us improve the second summer institute, as well as prepare us for the previously unanticipated changes caused by the pandemic disruptions. The most significant changes were the use of experienced teachers as the lead instructors, instead of using a university professor, and the course sequencing that taught both content knowledge and pedagogy topics daily, allowing teachers to reinforce the CS and CT topics that they learned in the morning in the afternoon through pedagogy. Moreover, our transition from an on-site, in-person to an entirely virtual experience was made easier due to the teacher leaders whom we recruited to teach, the use of a programming IDE that had a much smaller barrier to getting things started, and the redesign of collaborative activities put in place. Figure VII summarizes our SWOT analysis of the summer institutes.

VII. Conclusions and Future Work

In this paper, we first reviewed professional development (PD) programs for teachers in teaching computer science and computational thinking. Our review focused on the length of PD programs, their topics, and the programming languages used. We then presented our summer institutes, which are part of yearlong on-going PD programs. Our summer institutes are considered a type of long PD program, covering both CS and CT topics, as well as pedagogy. Finally, we carried out a SWOT analysis of the two summer institutes to identify strengths, weaknesses, opportunities, and threats. These insights have been valuable in helping us refine our summer institutes. We showed that our designs and summer institutes were viable and effective. Our future work includes extensive data analysis pertaining to our research investigations into how to better deliver PD programs for K-8 teachers, and how to better assess teacher learning and performance in PD and also in their classrooms. We are also currently recruiting a third cohort and in the process of planning for another summer institute, building on this SWOT analysis to further refine our plans. The third cohort's summer 2021 institute will again be all online.



Strengths	• Using a large instructional team to support and attend to teachers' needs
	• Instructional team was adaptive to teachers' need
	• Program had a positive impact on teachers' CT and CS knowledge
	• A virtual community was setup
	• Additional training during the subsequent academic year
	• Teachers helped each other understand difficult concepts
	• (2nd only) Use of JavaScript and JSFiddle.com made programming more approachable with their minimal setup
	• (2nd only) Due to online summer institute, there was effective use of breakout rooms through Zoom for group work
	• (2nd only) Due to online summer institute, there was ease of collaboration through screen sharing via Zoom (e.g., observation of other's code) facilitate peer-based learning
	• (2nd only) The 2-week of half-day schedule was adequate to cover all CS and CS pedagogy, also allowing for more effective learning at a slower pace than that in the 1st summer institute
	• (2nd only) Coupling the courses together helped motivate teachers
Weaknesses	• (1st only) Goals of instructor and teachers did not align
	• (1st only) CS course were not linked to CS pedagogy course immediately
	• Teachers were not prepared for transition to more difficult concepts
	• (2nd only) Explaining more intricate concepts was made difficult due to the online instruction
	• (2nd only) During breakout room discussions, facilitator-to-facilitator interactions would need to rely on a back channel to interact and coordinate at real-time
	• (2nd only) Online summer institute might have prevented the teachers from bonding as closely as they would in-person
	• (2nd only) Online summer institute prevented teachers from meeting with the instructional team in-person
Opportunities	• Use data collection platforms to solicit response and provide efficient in-time feedback
	• Collect feedback on the use of educational robots and tablets for future programs
	• Participants who did not have experience teaching CS had a chance to learn from peers and adapt lesson plans
Threats	• Not all CS concepts were covered in-depth
	• Collaboration among teachers made it challenging to facilitate individual work and promote individual accountability
	• Catering materials to very different experience and grade levels was a challenge
	• (2nd only) Due to online summer institute, instructional team was not able to observe whether the teachers used outside sources to aid them during the individual assessments at the end of the program
	• (2nd only) Due to online summer institute, instructional team was not able to ascertain whether teachers were paying attention—e.g., following along with programming—with their videos turned off and with also family responsibilities at home due to the pandemic

Figure VII. Summary of SWOT analysis of the two summer institutes

VIII. Acknowledgements

This was supported by the National Science Foundation under grant no. 1837476. All findings and opinions are those of the authors and not necessarily of the funding agency. The authors would like to acknowledge the instructors and teaching assistants and the teachers of the two summer institutes.

REFERENCES

[1] Ahamed, Sheikh Iqbal, et al. *Computational Thinking for the Sciences: A Three Day Workshop for High School Science Teachers*. ACM, 2010, pp. 42–46. *dl.acm.org*, doi:10.1145/1734263.1734277.

[2] Bell, Tim, and Jan Vahrenhold. "CS Unplugged—How Is It Used, and Does It Work?" *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*, edited by Hans-Joachim Böckenhauer et al., Springer International Publishing, 2018, pp. 497–521. *Springer Link*, doi:10.1007/978-3-319-98355-4_29.

[3] Bower, Matt, et al. "Improving the Computational Thinking Pedagogical Capabilities of School Teachers." *Australian Journal of Teacher Education*, vol. 42, no. 3, Mar. 2017, pp. 53–72.

[4] Brennan, Karen, and Mitchel Resnick. "New Frameworks for Studying and Assessing the Development of Computational Thinking." *Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada*, vol. 1, 2012, p. 25.

[5] Brown, Quincy, and Amy Briggs. "The CS10K Initiative: Progress in K-12 through 'Exploring Computer Science' Part 1." *Inroads*, vol. 6, 2015, pp. 52–53. *Semantic Scholar*, doi:10.1145/2803178.

[6] Desimone, Laura M., and Michael S. Garet. "Best Practices in Teachers' Professional Development in the United States." *Psychology, Society, & Education*, vol. 7, no. 3, Apr. 2015, p. 252. *DOI.org (Crossref)*, doi:10.25115/psye.v7i3.515.

[7] Ericson, Barbara, et al. "A Model for Improving Secondary CS Education." *ACM SIGCSE Bulletin*, vol. 37, ACM, 2005, pp. 332–336.

[8] Fancsali, Cheri, et al. "A Landscape Study of Computer Science Education in NYC: Early Findings and Implications for Policy and Practice." *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, ACM, 2018, pp. 44–49. *ACM Digital Library*, doi:10.1145/3159450.3159467.

[9] Goode, Joanna, et al. "Curriculum Is Not Enough: The Educational Theory and Research Foundation of the Exploring Computer Science Professional Development Model." *Proceedings of the 45th ACM Technical Symposium on Computer Science Education - SIGCSE '14*, ACM Press, 2014, pp. 493–98. *DOI.org (Crossref)*, doi:10.1145/2538862.2538948.

[10] Gwen, Nugent, et al. "The Effectiveness of Summer Professional Development for K-8 Computer Science Teachers." *Proceedings of Society for Information Technology & Teacher Education International Conference*, AACE, 2020, pp. 77-81.

[11] Gwen, Nugent, et al. "Impact of Summer Professional Development on K-8 Computer Science Teachers' Computer Science Knowledge and Attitudes", AERA Annual Meeting, Orlando, FL, 2021, April 9-22. [roundtable session]

[12] Hatlevik, Ove Edvard, et al. "Students' ICT Self-Efficacy and Computer and Information Literacy: Determinants and Relationships." *Computers & Education*, vol. 118, Mar. 2018, pp. 107–19. *DOI.org (Crossref)*, doi:10.1016/j.compedu.2017.11.011.

[13] Hill, Terry, and Roy Westbrook. "SWOT Analysis: It's Time for a Product Recall." *Long Range Planning*, vol. 30, no. 1, Feb. 1997, pp. 46–52. *ScienceDirect*, doi:10.1016/S0024-6301(96)00095-7.

[14] Kong, Siu-Cheung, and Andrew Chan-Chio Lao. *Assessing In-Service Teachers' Development of Computational Thinking Practices in Teacher Development Courses*. ACM, 2019, pp. 976–82. *dl.acm.org*, doi:10.1145/3287324.3287470.

[15] Lee, Irene A., et al. "Preparing STEM Teachers to Offer New Mexico Computer Science for All." *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, ACM, 2017, pp. 363–368. *ACM Digital Library*, doi:10.1145/3017680.3017719.

[16] Leyzberg, Dan, and Christopher Moretti. "Teaching CS to CS Teachers: Addressing the Need for Advanced Content in K-12 Professional Development." *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, ACM, 2017, pp. 369–374. *ACM Digital Library*, doi:10.1145/3017680.3017798.

[17] Liu, Jiangjiang, et al. "Making Games a 'Snap' with Stencyl: A Summer Computing Workshop for K-12 Teachers." *Proceedings of the 45th ACM Technical Symposium on Computer Science Education - SIGCSE '14*, ACM Press, 2014, pp. 169–74. *DOI.org (Crossref)*, doi:10.1145/2538862.2538978.

[18] McGee, Steven, et al. "Equal Outcomes 4 All: A Study of Student Learning in ECS." *SIGCSE '18 Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, Feb. 2018, https://ecommons.luc.edu/cs_facpubs/194.

[19] McGee, Steven, et al. "An Examination of the Correlation of Exploring Computer Science Course Performance and the Development of Programming Expertise." *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, ACM, 2019, pp. 1067–1073. *ACM Digital Library*, doi:10.1145/3287324.3287415.

[20] Milliken, Alexandra, et al. "Effective Computer Science Teacher Professional Development: Beauty and Joy of Computing 2018." *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, ACM, 2019, pp. 271–277. *ACM Digital Library*, doi:10.1145/3304221.3319779.

[21] Morreale, Patricia, et al. "Measuring the impact of computational thinking workshops on high school teachers." *Journal of Computing Sciences in Colleges*, vol. 27 no. 6, 2012, pp. 151-157.

[22] Novak, G.M., 2011. Just-in-time teaching. *New directions for teaching and learning*, *2011*(128), pp.63-73.

[23] Noone, Mark, and Aidan Mooney. "Visual and Textual Programming Languages: A Systematic Review of the Literature." *Journal of Computers in Education*, vol. 5, no. 2, June 2018, pp. 149–74. *Springer Link*, doi:10.1007/s40692-018-0101-5.

[24] Pollock, Lori, et al. "From Professional Development to the Classroom: Findings from CS K-12 Teachers." *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, ACM, 2017, pp. 477–482. *ACM Digital Library*, doi:10.1145/3017680.3017739.

[25] Salac, Jean, et al. "An Analysis Through an Equity Lens of the Implementation of Computer Science in K-8 Classrooms in a Large, Urban School District." *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, ACM, 2019, pp. 1150–1156. *ACM Digital Library*, doi:10.1145/3287324.3287353.

[26] Sengupta, Pratim, et al. "Integrating Computational Thinking with K-12 Science Education Using Agent-Based Computation: A Theoretical Framework." *Education and Information Technologies*, vol. 18, June 2013, pp. 351–80. *ResearchGate*, doi:10.1007/s10639-012-9240-x.

[27] Vogel, Sara, et al. "Visions of Computer Science Education: Unpacking Arguments for and Projected Impacts of Cs4all Initiatives." *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, ACM, 2017, pp. 609–614. *ACM Digital Library*, doi:10.1145/3017680.3017755.

[28] Wozney, Lori, et al. "Implementing Computer Technologies: Teachers' Perceptions and Practices." *Journal of Technology and Teacher Education*, vol. 14, no. 1, 2006, pp. 173–207.

[29] Yadav, Aman, et al. "Professional Development for CS Teachers: A Framework and Its Implementation." *Future Directions in Computing Education Summit*, 2013.