“It’s Hard Work” — K-12 Teachers’ Enacted Curriculum of Computational Thinking and Computer Science Integration

A Dissertation
SUBMITTED TO THE FACULTY OF THE
UNIVERSITY OF MINNESOTA
BY

Sarah Barksdale

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Dr. Cassandra Scharber, Advisor

May 2023

Copyright © 2023 by Sarah Barksdale

All Rights Reserved

ACKNOWLEDGEMENTS

"Thank you" does not feel adequate to convey the gratitude and appreciation I have for the mentorship, friendship, support, encouragement, generosity, and kindness I have received while on this journey. It doesn't escape me that in a dissertation full of *so many* words, I am still unable to come up with something that fully captures how deeply I have been touched by so many individuals during this adventure and the resulting gratitude I wish to express. For while this dissertation is a culmination of my personal doctoral journey, it is also a reflection of the impact of many folks who have touched my life and work. I (sadly) know not every person and moment will be captured here and my words don't begin to do my gratitude justice, but I still want to highlight and share additional thanks to some of the different people that make up the communities I have (had) the honor to be a part of.

Thank you:

To the brilliant **scholars** who have welcomed me, taught me, challenged me, and shared with me. While I have learned from so many during my time at the University of Minnesota, I especially want to highlight five professors who have played an integral part in creating and embracing my own scholarly identity. **Dr. Cassie Scharber** – you were my first contact at the University of Minnesota and you instantly inspired me to stay. I am so incredibly grateful to have had you as my advisor and for the many ways you showed up throughout my journey – from check-ins, encouragements, edits, advice, leadership, microwave bowl holders, Grand Marais recommendations, to so many incredible opportunities that shaped my own scholarship. Your support, words,

leadership, and belief that I could do this were instrumental in crossing the finish line. **Dr. Angelica Pazurek** – taking classes from you and teaching classes you developed profoundly changed who I am as a teacher and left me always inspired. The way you care for your students, the time and energy you put into your research, and the beautiful ways you put your ideas/care/findings into words are something I aspire to. You shine bright and I am forever grateful to have learned from and alongside you, and for the laughter, tears, camaraderie, giant writing pads, and wisdom you have shared with me. **Dr. Bhaskar Upadhyay** – I was drawn to your kindness from the moment you visited one of my classes, and then I had the honor and privilege to experience it firsthand in Nepal. Your practical advice, mentorship, willingness to grab coffee/tea, and opportunities to do what I love provided me with much needed hope and perseverance at the most perfect time. I treasure your wisdom and am so grateful to you. **Dr. Bodong Chen** – I have learned so much from your courses, the questions you have asked, and the insight you have shared. I always enjoyed bumping into you in the halls of LT, learning from you during class, and discussing ideas that left me invigorated. **Dr. J.B. Mayo** – You are someone that people are drawn to and I remember hoping I would get the opportunity to be in your orbit. I also saw an incredible model of pedagogy and care that I wanted to emulate while in your class. I treasure your friendship, authenticity, vulnerability, and mentorship. You are such a source of joy and I love the way you bring your whole self to everything you do – thank you for welcoming me in.

To the incredible **participants** of this study: **Mrs. Bee**, **Mr. Ozo**, and **Mr. Pi**. Your care and passion for your students and equitable computing experiences, as well as

your willingness to say “yes” made this dissertation possible. Thank you so much for welcoming me into your experiences and classrooms. It has been a true honor to learn from you and share your stories. You are beacons of hope.

To the amazing **friendships** that I owe to the University of Minnesota. Whether we walked/flew/roomed/presented/traveled/ate/laughed/learned/cried/wrote/complained or did it all together, I am so, so grateful for you. **Alex** and **Suzanne** – our writing sessions, walks on frozen lakes, cat masks, collaborative PD papers, brevity, laughter, honesty, bread making, and so much more have gotten me through this. Your students are so lucky to have you, as am I. So many of my UMN friends have walked beside me on this journey, paved the way, and/or kept me going (**Lana**, **Lavyne**, **Jeanna**, **Jennifer, Hui**, **Rukmini**, **LT colleagues**, and **Matt** to name a few) - I’m beyond grateful for each of you, the memories we’ve created, and the friendships we’ve formed.

For the life-changing **financial support** I’ve received from CEHD, C&I, LT, the Huebner Fellowship, and the P.E.O. Scholar Award. Thank you for your generosity and investing in me and my work.

To so many **loved ones** who inspire me, encourage me, and bring so much light to my life – it and you mean the world (**Katy**, **Trèza**, **Adriana**, **Kerstin**, **Hirsches**, **Andrea**, **Rachel**, and so, so many others).

To the **families** I have the delight of being a part of. The **Barksdales** – you have cared for me, encouraged and supported this journey, shared so much love with me, and always were willing to listen. **JWB, Jr**. – a special thank you for asking to be my first reader and your feedback. I couldn’t ask for a better bonus family and love you each so

much. **Serenia** – you are my brilliant sister, a phenomenal supporter, my role model, and a dear friend who has the most beautiful way with words. I treasure you and your family. **Dad** – "*Your absence has gone through me / Like thread through a needle. / Everything I do is stitched with its color*" (*Separation* by W.S. Merwin, 1993). I like to think I got my love of reading and learning from you, and I imagine this would have made you proud. Thank you for loving me while you were here. **Mom** – thank you for loving me from the very beginning and cheering me on even when it didn't make sense. I wouldn't be here without your selflessness, grace, love, and encouragement. I love you always.

To the family I have had the beautiful fortune of helping to create: a 1,000 thank you's wouldn't be enough. **John** – you have been my number one supporter since the beginning and have helped me in so many ways (from the little things like cleaning, supplying caffeine, making meals, giving me space or no space depending on what I needed, dreaming of the end with me, helping me get through Word formatting – to the big things like talking through ideas, moving, financial support, listening to me practice, and never doubting that the celebrating would come). I love you so much and appreciate the way you've been willing to embrace this season of life, learn alongside me at times, grieve with me, and celebrate with me. And **Anoka** and **Tiger Lily** – you have been with me since the start of this journey and your snores, cuddles, forced breaks/walks, joy, tail wags and purrs, and neediness have often given me the strength I needed to rest and/or keep moving forward. I couldn't be more proud to say the three of you are my family and woven so intricately into this journey.

I am so overwhelmingly grateful and privileged to have such incredible humans in my orbit currently and for those who have helped shape me during our past season(s) together. My words will never suffice, but I will keep trying – thank you.

DEDICATION

There is always light. If only we're brave enough to see it.

If only we're brave enough to be it.

- Amanda Gorman

To the little girls who refuse to give up, the teachers who empowered them to shine, and the communities who supported, encouraged, and lifted them up – may you each continue to be brave and shine bright for you are needed.

ABSTRACT

Integrating computational thinking (CT) and/or computer science (CS) into K-12 curricula has the potential to provide learners with early exposure to the possibilities of creation, engagement, and contributions within a technology-driven society. However, in order to weave CT and CS into their existing curricula and classrooms, K-12 teachers need training. While prior research has studied the design and immediate impact of computing-focused professional learning experiences for teachers (e.g., trainings, classes), there is little research on teachers' translation of these experiences into their own classrooms. This comparative multi-case study addresses this gap and contributes to the fields of computer science education and teacher education as it explores three K-12 Minnesota educators' enacted curriculum of integrating computing into their curricula and expands the research on CS/CT by illuminating teachers' conceptualization(s) of computing as it relates to their classrooms.

Utilizing the technological pedagogical content knowledge (TPACK) framework with the incorporation of computational thinking (TPACK-CT) as a lens and conceptual framework, this study highlights the pedagogical and technological strategies for how a K-5 STEM specialist, a fourth grade teacher, and a high school computer science teacher in Minnesota integrated CT/CS into content areas. To answer the research questions an inductive, constant comparative method and qualitative content analysis were used on the data (surveys, observations, interviews, and reflections) to construct three rich and detailed case portraits, followed by a cross-case analysis. Four themes (*PD as a bridge for buy-in, integration, and permission; importance of conceptualizing CT for their*

classroom; *(missing) connections in student-centered, collaborative, and balanced pedagogical strategies*; and *the (un)plugged fun and supportive role of technology*) are discussed in an attempt to capture and make sense of the common threads woven throughout the case portraits and cross-case findings, while also providing implications and insight for moving forward in practice and research.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS i

DEDICATION vi

ABSTRACT vii

TABLE OF CONTENTS ix

LIST OF TABLES x

LIST OF FIGURES xi

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: REVIEW OF LITERATURE 10

CHAPTER 3: METHODOLOGY 49

CHAPTER 4: MRS. BEE 78

CHAPTER 5: MR. OZO 97

CHAPTER 6: MR. PI 124

CHAPTER 7: CROSS-CASE FINDINGS 147

CHAPTER 8: DISCUSSION AND SUMMARY 173

REFERENCES 196

APPENDICES 211

LIST OF TABLES

Table 1: Overview of Respondents to Recruitment Survey..... 55

Table 2: Participants..... 57

Table 3: Alignment of Research Questions and Data Sources 65

Table 4: Timeline of Study and Data Collection 66

Table 5: Totals of Pages, Time, Images, and Artifacts Accumulated During Data Collection..... 70

Table 6: Overview of Data Analysis Stages 71

Table 7: A Summary of the Challenges of and Helpful Supports* for Computing Integration in K-12 Education that Emerged During Cross-Case Theme Construction. 176

LIST OF FIGURES

Figure 1. A Barefoot Computing poster 18

Figure 2. This study's operational definition of CT (CSTA & ISTE, 2011) 19

Figure 3. An overview of computing education in Minnesota. 39

Figure 4. The TPACK Framework (Kurt, 2018) 44

Figure 5. A visualization for the TPACK-CT framework 45

Figure 6. An illustrated conceptualization of a researcher kaleidoscope 58

Figure 7. Mrs. Bee's introductory ID card 78

Figure 8. Mrs. Bee's classroom arrangement. 80

Figure 9. Student work examples 86

Figure 10. Materials to introduce a lesson 88

Figure 11. Mrs. Bee helps students as they gather around to see a students' project 89

Figure 12. Mrs. Bee introduces Beebots to the class 91

Figure 13. Materials and lessons to introduce agricultural engineering 93

Figure 14. Mr. Ozo's introductory ID card 97

Figure 15. Students work on their Scratch projects in Mr. Ozo's classroom before school begins 99

Figure 16. Mr. Ozo's classroom arrangement 100

Figure 17. Game and puzzle examples used by Mr. Ozo. 109

Figure 18. Mr. Ozo shares a spreadsheet of data he would have used as an engineer 114

Figure 19. The code behind a fractal program in Scratch 120

Figure 20. Mr. Pi's introductory ID card 124

Figure 21. Mr. Pi’s classroom arrangement 127

Figure 22. Students and Mr. Pi look at some of the available technologies and supplies in Mr. Pi’s classroom 140

Figure 23. Students discuss one of the district's equity statements 143

Figure 24. A bar graph showing how the participants conceptualized computational thinking 161

Figure 25. A bar graph illustrating confidence with computing 163

Figure 26. A bar graph illustrating confidence with teaching computing 164

Figure 27. A bar graph illustrating beliefs around computing and education. 166

Figure 28. A visual representation of PD as a bridge 178

Figure 29. A visualization of CT content knowledge conceptualized differently 184

Figure 30. A visual of (Missing) Connections 187

Figure 31. The balance of structure and fun when using (un)plugged technology 189

CHAPTER 1: INTRODUCTION

As technologies continue to be more interwoven into our daily life, it is critical that people have the knowledge and skills necessary to understand, critique, use, troubleshoot, and create with these technologies (Czerkawski, 2015). Many educators and researchers have proposed that computational thinking (CT) and/or computer science (CS) education are avenues for developing K-12 students' 21st century skills, which are needed for imminent technological challenges, and have identified CT/CS as an emerging digital literacy (Guzdial, 2019; Scharber et al., 2021; Vee, 2018). For instance, the International Society for Technology in Education's (ISTE) "Standards for Students" (www.iste.org) not only includes CT as a standard to help students "thrive in a constantly evolving technological landscape," but they also released specific CT competencies in 2018 to "prepare students with the skills needed to solve problems of the future" (ISTE, 2018, 2020). There is now worldwide interest around integrating computational thinking and computer science into K-12 education and curriculum (Alqahtani et al., 2022; Barr & Stephenson, 2011; Bower et al., 2017; Goode et al., 2012; Jocius et al., 2021; Merino-Armero et al., 2022). In response to initiatives calling for all students to have opportunities to learn the concepts and skills used in computer science and/or computational thinking, multiple countries (e.g., Australia, Ireland, United Kingdom, and United States, etc.) are working to find ways to prepare teachers to integrate these skills into their existing curricula (Balanskat & Engelhardt, 2015; Bower et al., 2017; Hsu, 2019; Mouza et al., 2017; Yadav et al., 2018, 2022).

Despite the call for integrating computational thinking and computer science into K-12 education and curriculum, there are many disparities in who has access to these opportunities (Barksdale et al., 2022; Code.org et al., 2022). Research and history highlight how computer science has disproportionately left out students with disabilities and students of color by creating systemic and structural barriers, such as tracking, lack of access, lack of culturally relevant curriculum and equitable pedagogical strategies, un(der)prepared teachers, prerequisites, and other implicit and explicit obstacles. For example, the state of Minnesota is demonstrative of this opportunity gap as it currently ranks last in the country for number of high schools offering computer science (Code.org et al., 2022; CSforAllMN, 2020a).

Many researchers and educators have called for integrating computing into the K-12 curriculum as a way to begin addressing the disparities of who has access to and participates in computing. Mouza et al. (2018) point out that when studying computing, it is difficult to separate CS concepts from other computing-related areas like digital citizenship and computational thinking, so integrating CT or CS may (un)intentionally incorporate the other, which is why this study explores both under the umbrella of "computing." First, though, it is helpful to understand their differences. While computer science utilizes and automates computational thinking concepts/processes through computing tools and technologies, computational thinking is more interdisciplinary in nature as it focuses on the process itself (e.g., abstraction) and does not require computing tools to be learned (Barr & Stephenson, 2011; Wing, 2012). Wing (2012) explains that CT "is the thought process involved in formulating a problem and

expressing its solution in a way that a computer - human or machine - can effectively carry out…[it] is what comes before any computing technology" (p. 7). Computational thinking is an avenue to bring computing ideas into K-12 classrooms that benefits all students because it is interdisciplinary and applicable to more than computer science; it provides an accessible entry point for teachers to integrate it into their curriculum; and it does not require technology to learn/practice it (Yadav et al., 2018). Embedding computing content into existing K-12 curriculum ensures learners have early access to CT/CS in elementary school, a time when interest and computing identity development are key for youth (Peterson et al., 2019).

Not only does incorporating CT into elementary school allow for early promotion of interest in computing during identity development before learners begin choosing career paths (Tai et al., 2006), but it also increases the potential to reach *all* students (Ketelhut et al., 2020). Other benefits researchers often note when advocating for integrating computing within other subject areas include being able to make more "real-world" connections, contributing to the creation of an innovative CS curriculum, appealing to girls, increasing access, being more sustainable, and expanding problem-solving (Denning et al., 2017; Miller, 2020; Ozturk et al., 2018; Sahami et al., 2013). Since creating new subjects and classes is often not doable in public schools, particularly in elementary school, integrating computing into core subjects of K-12 education, like mathematics, science, and language arts, is more sustainable as it works within the complexities of K-12 systems, honors the time and resource constraints within these systems, and provides students opportunities to see computing within other discipline

contexts (Yadav et al., 2019). For example, math teachers may use abstraction (CT concept) as a way to understand fractions or science teachers may use debugging (CT concept) as they experiment to better understand different science concepts.

Teachers need training and support to both understand and equitably integrate computing into their curriculum (Fields et al., 2019) in order for students to have early access to computational thinking during their K-12 education. While much research has been done to explore the design of specific professional learning opportunities for K-12 teachers and on ways to incorporate computing into teacher education (e.g., trainings, workshops, cohorts, academic courses), there is a need for research that investigates what actually occurs in the classroom beyond these learning experiences - the enacted curriculum (Bower et al., 2017; Kong et al., 2023; Mannila et al., 2014; Yadav et al., 2014). This dissertation study addresses this gap as it explored how three Minnesota educators integrate computing into their curriculum.

Study Overview

The purpose of this study was to look at teachers' enacted curriculum of integrating computing, particularly computational thinking (CT) and computer science (CS), into their classroom; the focus was on identifying what actually happens in the classroom beyond participation in professional learning (e.g., a workshop, coursework in their degree program, etc.). Utilizing the technological pedagogical content knowledge (TPACK) framework (Koehler et al., 2013) with the incorporation of computational thinking (TPACK-CT), this comparative multicase study explored how three Minnesota educators integrated CT/CS into content areas, or their enacted curriculum.

In order to illuminate what teachers' TPACK-CT looks like when enacted in the classroom, data (observations, interviews, artifacts, reflections, and a survey) was aligned with each research question and collected during the 2021-2022 school year. Using an inductive, constant comparative method and qualitative content analysis, I created detailed case portraits of each participant's enacted curriculum to answer the first research question and its sub-questions. To answer the second research question, a cross-case analysis was done.

Research Questions

- What does teachers' TPACK-CT look like when enacted in the classroom? *[technological pedagogical and content knowledge related to CT, TPACK-CT]*
    - How do teachers conceptualize computational thinking knowledge in their classroom? *[content knowledge, CK-CT]*
    - What pedagogies are being enacted in the classroom when teaching CT/CS? *[pedagogical knowledge, PK-CT]*
    - What technologies, if any, do in-service teachers utilize for teaching CT/CS? *[technological knowledge, TK-CT]*
- How do teachers' TPACK-CT compare and contrast across cases?

By answering these research questions, this multi-case study contributes to the fields of computer science education and teacher education as it explored how educators situated in a state where computing does not have its own set of standards go about integrating computing into their curriculum, and expands the research on CT by illuminating teachers' conceptualization(s) of CT situated within their contexts. Finally, the study

shares the nuanced nature and approaches to integrating computing, resulting in additional insight into strategies for integrating computing in K-12 classrooms.

Definitions

I use several terms in this dissertation, and it is important to understand how I conceptualize them. The terms are organized by the order they are discussed in the review of literature (Chapter 2). The definitions are drawn from both scholars in the education research field and leading organizations such as the Computer Science Teachers Association (CSTA, www.csteachers.org) and the International Society for Technology Education (ISTE, www.iste.org). It is important to note that since this study focused on the teachers' experiences, the teachers themselves may have interpreted this terminology differently depending on the context(s). For instance, computational thinking, computer science, and computing were sometimes differentiated by the teachers and other times were used interchangeably throughout the study because that reflects the reality of their experiences (Ozturk et al., 2018).

- Computer Science (CS): This study utilizes CSTA's definition to conceptualize computer science as it allows room to capture both the concepts and practices of CS. Computer science is "the study of computers and algorithmic processes, including their principles, their hardware and software designs, their implementation, and their impact on society" (CSTA, n.d.).
- Computational Thinking (CT): Drawing on CSTA and ISTE's definition (2011), CT is conceptualized as a problem-solving process that includes concepts (such as decomposition, algorithmic thinking, abstraction, data collection, automation,

etc.) and practices that encourage dispositions such as confidence in dealing with complexity, persistence in working with difficult problems, communication with others, and more.

- Computing: The following definition that encompasses both CT & CS is used, "Any goal-oriented activity requiring, benefiting from, or creating computers" (ACM & IEEE, 2005, p. 9). Unless articulated, the use of computing in this study does not include foundational computer literacy and skills (e.g., typing, how to log in, etc.).
- Integration: A pedagogical decision to connect concepts (e.g., CS or CT) with one or more content areas, and/or to the "real world" or world beyond the classroom (Mohr & Welker, 2017) - in other words, it is not only teaching a subject like CT/CS in isolation.
- Enacted curriculum: The operationalization of the intended curriculum (the standards or the learning goals) in the classroom (Nolet & McLaughlin, 2000), including teacher beliefs, knowledge of pedagogy and curriculum, access to resources, understanding of student needs, and local context (Remillard & Heck, 2014).
- TPACK-CT: The technological pedagogical and content knowledge (TPACK) used for the integration of computational thinking. Specifically, the computing tools, terminology, practices, and dispositions for cultivating understanding of CT (TK-CT), making connections between CT and existing content (CK-CT), as well as the pedagogical methods used to make these connections (PCK-CT).

Overview of Chapters

This first chapter shares a brief, high-level overview of the state of computing education, including why this emergent digital literacy is of interest to educational scholars and researchers. The need for computing education to be accessible and equitable is highlighted, as well as the resulting research on designing and implementing professional learning experiences to prepare teachers for integrating computing into K-12 education. Finally, my study is situated amidst the computing education research landscape to showcase how exploring the enacted curriculum of integrating computing framed by the TPACK-CT framework provides insight into both teachers' conceptualization(s) of computing as well as what is happening in classrooms after PD experiences have concluded, and shares pedagogical strategies from educators integrating computing into their classrooms.

Chapter 2 reviews the literature around computing education, further defines the nuances of computer science and computational thinking, and shares the history and current realities of equity in computing. Next, I highlight some of the research on integration as well as research on computing integration, and an overview of intended and enacted curriculum. Last, I share more about the computing education landscape in Minnesota and how my conceptual framework, TPACK-CT, acts as a lens to view and frame/organize my study.

The third chapter discusses why a comparative multicase study approach was chosen for this study, as well as how the state and participants were selected. Next my role as the researcher, or my "researcher kaleidoscope," is elaborated upon. Finally, I

describe my data collection methods and analysis processes, as well as the validity strategies I took to ensure my study drew meaningful and accurate conclusions.

Chapters 4, 5, and 6 share the resulting case portraits of Mrs. Bee, Mr. Ozo, and Mr. Pi - three Minnesota educators. Mrs. Bee is a K-5 STEM Specialist at a suburban elementary school; Mr. Ozo is a fourth grade teacher at a rural elementary school; and Mr. Pi is a high school computer science teacher at a city community technical college. These chapters share the enacted curriculum of each of these teachers organized by the TPACK-CT framework to better answer the first research question and its sub questions. Chapter 7 is the cross-case analysis that answers the second research question as it shares the thematic findings that are visible across the three cases.

The final chapter of this dissertation, Chapter 8, summarizes the research, aligns my findings within existing literature, and uses four themes to guide the discussion of the implications of my findings. The study's limitations and recommendations for future research are shared and the chapter concludes with a call to action.

CHAPTER 2: REVIEW OF LITERATURE

This chapter begins with a brief introduction followed by an overview of computing education, specifically computer science and computational thinking. Some of the complex history and the role of equity in computing is shared since it is such an integral part of computing education. Next, a brief overview of curriculum integration sets the stage for looking at the integration of CS and CT, and is followed by reviewing K-12 teacher preparation for integrating CS/CT. Finally, the chapter concludes with defining the concepts of intended and enacted curriculum as well as an explanation of the conceptual framework (TPACK-CT) used in this study.

In response to the rise of K-12 computer science (CS) standards and computational thinking (CT) being added to already existing standards, the call for CS/CT to be connected to multiple disciplines, and the ask for K-12 teachers to teach and/or integrate CS/CT into their curriculum, researchers are designing specific methods and experiences where educators can develop and grow their knowledge of computing (including CT) and pedagogical strategies. Despite clear differentiation between CS and CT definitions and diverse applications in K-12 education, the terms are frequently not defined and/or differentiated in common use or within academic literature (Grover & Pea, 2013; Northrup et al., 2022). For example, Borowczak and Burrows (2019) begin to capture their interconnectedness when they defined computer science as “the science of problem solving within a computational context” (p. 1) and framed CT as a connection between CS and the visible practice of computer programming. To match and reflect the

realities of computing education in the United States, this study, including its literature review, explored and expanded upon research related to both CS and CT.

In their work with new-to-CS teachers, Northrup et al. (2022) explored how their findings of a one-year CS training skills certificate cohort program could "serve as a bridge between the mandates created by education leaders and the actual experiences of the participating teachers tasked with delivering the curriculum" (p. 353). They found that the cohort of thirteen teachers believed that CS is the future, noted that a pedagogical shift is required to teach CS since students may know more than the teacher, struggled with learning foundational concepts (including CT), and required well-constructed PD and support to teach CS effectively. Mouza and colleagues (2017) call for further exploration of the pedagogical approaches used to infuse CT knowledge and skills as well as the need for the distilling of this knowledge. Vivian and Falkner (2019) point out that with the emergence and evolution of CS curricula, more research is needed "to better understand the interrelationships and best practices in CS-related TPACK" as teachers navigate teaching new CS content supported through pedagogy and technology (p. 147) because "it is necessary that professional development providers support teachers to navigate these various knowledge domains with the CS context" (p. 149). Yadav and colleagues (2014) also emphasize the importance of developing teachers' CT in the context of the content they already teach, and the need to examine "how teachers would actually implement computational thinking in their classrooms after being provided with professional development" (Yadav et al., 2014, p. 5).

By exploring how teachers in Minnesota conceptualize CT and the pedagogical choices enacted to weave it into their classrooms, this study not only illustrates specific strategies for integrating CT/CS, but also illuminates what happens in the classroom beyond CS/CT-focused professional learning experiences.

Computing Education

With the prevalence of computers and technology in the world today and in everyday life (K-12 Computer Science Framework, n.d.), there has been a push for computing to be interwoven into education so that people are educated to "utilize that technology most effectively to the benefit of mankind" (Tucker, 2003, p. 3). Denning (2007) explains that computing is a natural science and that computation was present before computers were invented, so it is not just isolated to computer science. He asserts, "computing is…the study of information processes, natural and artificial" (p. 18). Computing is a broad term in educational research that encompasses not only computer science and programming (or coding), but also the phenomena surrounding computing (e.g., privacy, security, ethics, etc.), including computational thinking:

> In a general way, we can define computing to mean any goal-oriented activity requiring, benefiting from, or creating computers…The list is virtually endless, and the possibilities are vast. Computing also has other meanings that are more specific, based on the context in which the term is used. (ACM & IEEE, 2005, p.9)

Within the context of K-12 education, computing is used as "an umbrella term that houses computer science, technology or digital literacy, digital citizenship, and all

concepts and practices related to computers and technology," and is often used interchangeably with computer science to lump these different concept areas together (Guo, 2022, p. 167). Yet while technology/digital literacy and digital citizenship are important, they are more focused on using computer technologies whereas computer science focuses on understanding why and how computers work (K-12 Computer Science Framework, n.d.). As computer science education has continued to grow, multiple professional organizations (i.e., the Association for Computing Machinery (ACM), the Association for Information Systems (AIS), the Institution of Electrical and Electronics Engineers Computer Society (IEEE-CS), and the Computer Science Teachers Association (CSTA)) have attempted to clarify the concepts of computing and computer science through the building of education standards (Guo & Ottenbreit-Leftwich, 2020). Guo and Ottenbreit-Leftwich (2020) explain that there are two stages of computer science scholarship: one stage focuses on general computing education at the post-secondary level and the other looks at the K-12 levels and is usually termed computer science education (p. 1). While this study will use the term computing to encompass both computer science and computational thinking, I first examine some of the nuances of both computer science and computational thinking at the K-12 levels as well as the intended curriculum surrounding computing education.

K-12 Computer Science

The Computer Science Teachers Association K-12 CS Education Glossary defines computer science as "the study of computers and algorithmic processes, including their principles, their hardware and software designs, their implementation, and their

impact on society" (CSTA, n.d.). The K-12 Computer Science Framework also adds what CS is not: "computer science is often confused with the everyday use of computers and computer applications, such as learning how to access the Internet and use digital presentation software" (n.d.). Instead they lay out a framework for CS concepts (what students should know) and practices (what students should do). Their vision for computer science is cross-disciplinary and involves using CT to solve problems in order to become creators of computing technologies.

The emergence of standardized policy tools (e.g., standards, frameworks, and guidelines) for K-12 computer science also provides insight to the conceptualization of CS as they outline more detailed, measurable expectations of what CS should look like in the K-12 education context (Falkner et al., 2019). In the United States (U.S.), computer science standards vary by state; while there is no universal set of standards in the U.S., the Computer Science Teacher Association (CSTA) developed K-12 CS Standards that can be either implemented as-is or built upon in states (Guo & Ottenbreit-Leftwich, 2020). In contrast, postsecondary education does not have a set of standards but instead frames and approaches CS differently as the goal is usually to become a computer scientist in the workforce, so computer science curricula moves from theory to application scope (Guo, 2022).

While there is often agreement on the need for CS education amongst organizations, governments, and individuals, their reasons for it often vary and are reflected in the variety of initiatives and approaches to integrating it into the public school system (e.g., CSforAll, Expanding Education Computing Pathways, school

initiatives, state departments of education approach to CS standards, etc.). Scholars have tried to capture these emergent rationales for bringing computer science into the compulsory education system. In their review of how computer science was positioned in several countries' curricula, Fluck and colleagues (2016) identified three rationales: economic, social, and cultural. They explain that the economic rationale supports both needing computer scientists to be competitive in today's technology-driven world but also meets the need for "computer science-enabled professionals in all industries to support innovation and development" (p. 41). In comparison, the social rationale emphasizes the value of being technology creators and producers to provide people with power to choose their role in society. Lastly, the cultural rationale positions people to be able to drive cultural change. Within the U.S., Vogel and colleagues (2017) identify seven core values that are reflected in the reasons for and approaches to K-12 CS education: equity and social justice; competencies and literacies; citizenship and civic engagement; technological, social, and scientific innovation; economic and workforce development; school reform and improvement; and personal agency, joy, and fulfillment. Ultimately, while their reasons may differ, many believe that computer science is a necessary literacy that students will not only use in their daily lives but also in their future careers - and not just those careers within science, technology, engineering, and mathematics (STEM) fields (Guzdial, 2019; K-12 Computer Science Framework, n.d.; Scharber et al., 2021; Vee, 2018).

Despite the growth of computer science in K-12 education, CS teachers come from various backgrounds and experiences resulting in being/feeling un(der)prepared to

teach it due to gaps in their knowledge/skills in teaching and/or CS (Sadik, 2017; Zhu & Wang, 2023). In their review of international models of CS teacher education, Yadav et al. (2022) identified some of the common challenges CS teachers face such as being the only CS teacher in the school and not identifying themselves as CS teachers because their background/training is in another subject area. Other research (e.g., Codding et al., 2021; Hubbard, 2017; Northrup et al., 2022; Veletsianos et al., 2016; Zhou et al., 2020; Zhu & Wang, 2023) demonstrated that professional learning experiences can be beneficial for developing CS content knowledge, self-efficacy, and pedagogical strategies. Scholars also highlight the possibilities of integrating CS across subject areas for meaningful learning and connections (Vivian & Falkner, 2019) and the benefits of including it in primary education (El-Hamamsy et al., 2021).

Core topics in computer science also implicitly and explicitly exercise computational thinking skills, which is another domain of computing that not all CS teachers feel proficient in. In Sadik's study (2017), one of the major needs CS teachers shared was ideas for embedded CT in their curriculum. CS and CT are intricately intertwined within computing because computer science is a foundational piece of CT - it is part of what separates CT from other problem-solving methods (Duncan et al., 2017, p. 2). Despite their overlaps, CS teachers may have limited knowledge of CT which creates a barrier (Sadik, 2017). Therefore, it is also important to understand computational thinking learning as it is not dependent on computers and can be taught both within and outside of computer science.

K-12 Computational Thinking

The concept of computational thinking can be traced back to Papert (1980) but has gained more attention after Wing (2006) defined it over twenty years later. A few years later, Wing (2010) and her colleagues provided a revised definition stating that CT is "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent." Selby and Woollard (2013) proposed another definition based on a synthesis of 39 published papers:

> A cognitive or thought process that reflects: the ability to think in abstractions, the ability to think in terms of decomposition, the ability to think algorithmically, the ability to think in terms of evaluations, and the ability to think in generalizations. (p. 5)

More recently, Shute et al. (2017) reviewed over 45 papers relevant to computational thinking and proposed the following definition of CT: "The conceptual foundation required to solve problems effectively and efficiently (i.e., algorithmically, with or without the assistance of computers) with solutions that are reusable in different contexts)" (p. 151). Barefoot Computing, an organization funded by the Department of Education in the UK in 2014 to prepare primary school teachers for computing curriculum, simplifies computational thinking to being "all about solving problems effectively - with or without a computer" (Barefoot Computing, n.d.). They break it down further into six concepts (logic, algorithms, decomposition, patterns, abstraction, and

evaluation) as well as five approaches or practices (tinkering, creating, debugging, persevering, and collaborating) (see Figure 1).

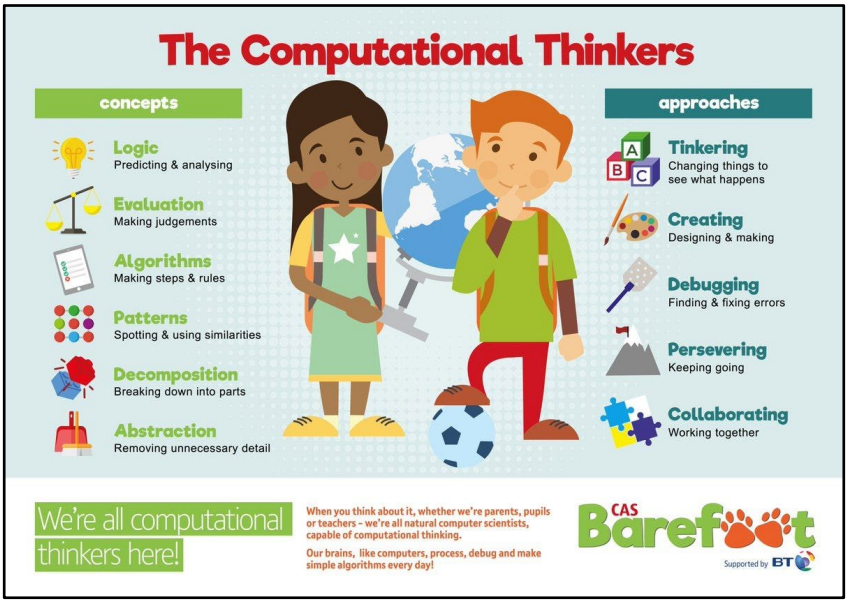


Figure 1. A Barefoot Computing poster. The poster showcases the six practices and five approaches to being computational thinkers.

While there remains a debate on the exact definition of CT, there is agreement that it is rooted in computer science and is viewed as "a universally applicable attitude and skill" (Wing, 2006, p. 33) that involves analytical thinking (Bers, 2018) and the process of problem-solving (ISTE, 2020). Unlike CS, there are no standards written just for computational thinking, although it is embedded into other standards (Coenraad et al., 2022). This study conceptualizes computational thinking as the CT concepts and competencies identified by ISTE and the Computer Science Teacher Association (CSTA), which include problem decomposition, algorithmic thinking, abstraction, data collection, automation, confidence in dealing with complexity, persistence in working with difficult problems, and more (CSTA & ISTE, 2011) (see Figure 2).

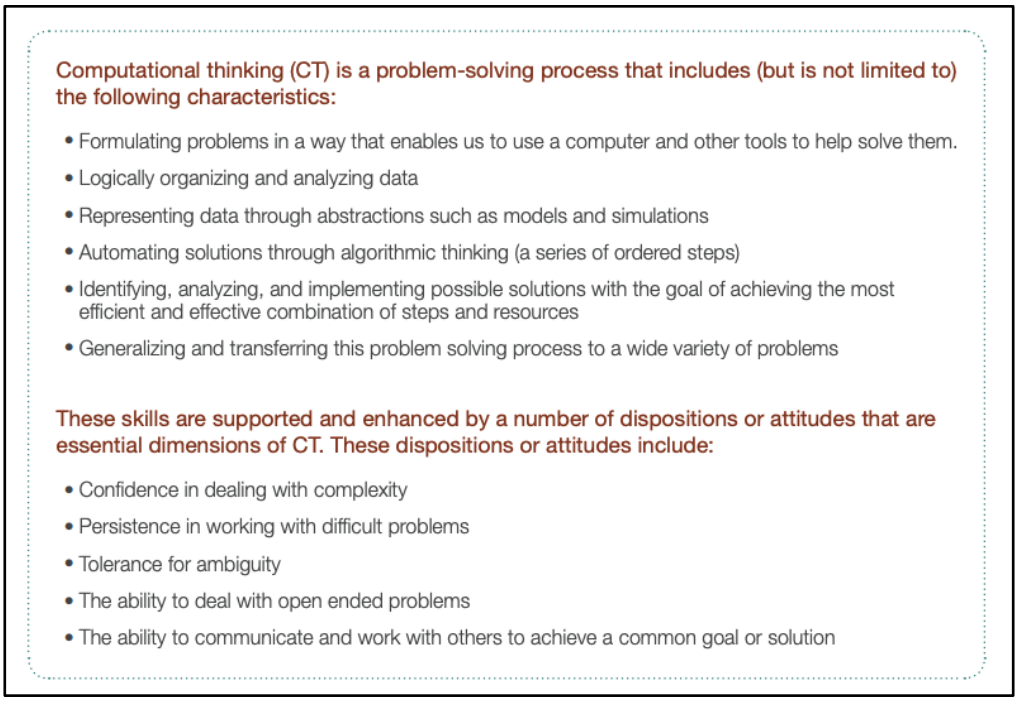


Figure 2. This study's operational definition of CT (CSTA & ISTE, 2011).

While CT is often used and/or taught in CS, the two are not equivalent and computational thinking does not require programming to be taught/learned (Fields et al., 2019; Voogt et al., 2015). For instance, some educators have created unplugged activities, or activities that use no technology, to teach computational thinking (e.g., Delal & Oner, 2020; Looi et al., 2018; Mensan et al., 2020; Merino-Armero et al., 2022). Research also shows that CT can be taught and learned in both formal and informal learning settings (Jocius et al., 2021; Voogt et al., 2015) in addition to being integrated into K-12 curriculum (Grover & Pea, 2013; Merino-Armero et al., 2022). Yadav and colleagues (2019) agree that CT is central to computer science, but also that CT is a "skill for students to learn within the context of other disciplines" (p. 2). In Moudgalya and colleagues' (2021) teacher development workshop for pre- and in-service teachers to integrate CT across subject areas, they found that teachers believed CT will be a

"gateway" for students to pursue CS rather than seeing coding/programming as the only way to develop CT skills.

Educational research on computing education often looks at the interdisciplinary applications of both computer science and computational thinking. In order to explore this more, K-12 teachers are often encouraged to integrate CS and/or CT into curriculum - especially as more computing-related standards are woven into states' existing content standards. Therefore, it is important to understand the history of equity in computing as it is often referenced in standards and sometimes serves as a catalyst for the integration efforts of CS/CT.

Equity & Computing (Education)

While this study did not explicitly focus on the creation or implementation of equitable experiences or curricula for students (e.g., Barksdale et al., 2022; Mouza et al., 2016) nor on the implementation or impact of professional development solely for equitable teaching practices (e.g., Codding et al., 2021; Goode et al., 2021), equity issues and insights are woven throughout this dissertation and its findings. Flapan et al. (2021) state, "Our nation's history of racism and other forms of discrimination and bias is manifested today in structural school inequalities that result in unfair and unequal learning outcomes" (p. 3) and computing education is one example of the reproductions of these inequalities. How teachers conceptualize computing, including their beliefs about computing, and their pedagogical practices impact students' computing education experiences (Koshy et al., 2022; Vakil & de Royston, 2019; Zhou et al., 2020). Therefore having an understanding of the historical role of equity in computing, as well as some

familiarity with the current research on equitable K-12 computing education will be helpful in molding the lens the reader views this dissertation and its findings through.

The history and role of inequities within computing have been featured and explored both in popular culture via documentaries and movies (e.g., *Coded Bias* reveals the biases embedded within artificial intelligence; *Code: Debugging the Gender Gap* looks at the lack of women and minority software engineers; and *Hidden Figures* highlights the Black women mathematicians, or human computers, who worked for NASA and were integral in the Space Race) and books and magazine articles (*Race After Technology: Abolitionist Tools for the New Jim Code* by Ruha Benjamin explains how emerging technologies are not neutral and can reinforce white supremacy and deepen social inequity; *Technically Wrong: Sexist Apps, Biased Algorithms, and Other Threats of Toxic Tech* by Sara Wachter-Boettcher shares the oversights, biases, and ethical issues within technology; and the 2019 feature "The Secret History of Women in Coding" in the *New York Times Magazine* explored how the gender balance in computer programming became unbalanced over time), as well as in academic inquiry and research reports (e.g., *Stuck in the Shallow End of Computing*, Margolis, 2010/2017; and *Moving Towards a Vision of Equitable Computer Science*, Koshy et al., 2022). Some of these explore the history of computing, particularly focusing on the decline of women in CS over time, while others focus on the repercussions of not having multiple, diverse perspectives creating and designing the technology used today and the resulting replication of societal injustices. Others explore the "pipeline" metaphor to see how the disparate numbers of girls and students of color participating in and/or passing advanced

U.S. placement (AP) CS courses in schools can also be seen in the low numbers of women and people of color in computing fields/jobs compared to other STEM fields.

Equity is a key piece of computing history and is woven (both implicitly and explicitly) across K-12 computing education - in fact even the K-12 CS Framework includes equity as a key theme across its concepts and practices, and Standard 2 of the CSTA CS Standards for teachers is "equity and inclusion." In addition, much of the rhetoric and growth of research on and approaches to computing education are in response to the historical displacement of women and lack of representation of people of color that continues to persist today (Scharber et al., 2021; Yadav & Heath, 2022). While scholars have begun to reject seeing the value of computing as a pipeline for historically marginalized people to enter the computing or STEM workforce (e.g., Vakil & de Royston, 2019; Yadav & Heath, 2022), much of the literature still focuses on broadening participation in computing through increasing access to computing, as well as providing robust, equitable experiences for students (Gretter et al., 2019; Koshy et al., 2022). In their 2022 report on the landscape of U.S. CS education, Koshy et al. (2022) say that in order "to ensure that historically-marginalized students receive opportunity and access to quality CS education, experience success in CS classrooms, [and] are affirmed and cared for in CS classrooms" then the following items should be present in equitable CS classrooms: "valuing student agency and voice, including community voice, equipping teachers and schools with resources and trainings to deliver quality instruction, supporting all students to disrupt racism, inequity, and other systemic barriers" (p.5). Studies have also shown that the following strategies can help create equitable computing

environments: incorporate students' identities and interests, utilize engaging and culturally-relevant curriculum, include collaboration and group work, use connected learning strategies, opportunities for role models/exposure to fields, create connections, allow for creativity, and explore and include social justice issues that are prevalent in today's society via computing (Barksdale et al., 2022; Blikstein, 2018; Goode et al., 2018; Gretter et al., 2019; Peterson et al., 2019).

Recent literature on computing education often focuses on ways to center equity in computing curricula and pedagogical strategies that teachers can employ to provide students with equitable experiences. For example, Yadav and Heath (2022) argue that CS teachers should address racialized and patriarchal biases in their curriculum via a participatory and community-centered approach. Vakil and de Royston (2019) drew on video data of an equity-oriented CS classroom to advocate for prioritizing the cultivation of positive student relationships amidst curricular innovations that bring in ethical and sociopolitical issues surrounding CS and its role in society. In their CT integration workshops with pre- and in-service teachers, Moudgalya et al. (2021) intentionally included discussions around how CT can promote equity, as well as examples of ways for students to use computing to express their whole selves and cultural backgrounds.

Research is also beginning to look at how school leadership can be included in order to bring about more equitable, scalable, and sustainable CS education (Flapan et al., 2021; Peterson et al., 2021), as well as the development of a framework to assess equity in four key components of CS education: **C**apacity for, **A**ccess to, **P**articipation in, and **E**xperience of equitable CS education (CAPE) (Fletcher & Warner, 2021). Equity is also

often one of the drivers for computing integration into elementary curriculum, as well as into other subject areas, as it has the potential to reach all students (Coenraad et al., 2022; Ozturk et al., 2018).

Integration

Integration was discussed as early as the 1890s and early 1900s when John Dewey and Francis Parker "established the idea of an integrated curriculum as a vital part of effective pedagogy" and now integration is often included in discussions regarding how to combine subjects to meet cross-curricular objectives (Hinde, 2005, p. 106). Beane (1995) asserts that curriculum integration embodies more than "cosmetic changes or realignments in lesson plans across various subject areas," (p. 616) and instead "has its own theories of purpose, knowledge, and learning and is able to stand on those without the necessity of standing on the corpse of the separate-subject approach" (p. 619). Beane (1995) radically argues that curriculum integration is not to be confused with approaches that are merely just moving beyond separate subjects nor themes that are identified just because they exist in current subjects, rather he posits that curriculum integration:

> begins with the identification of organizing themes or centers for learning experiences. As previously noted, the themes are drawn from real-life concerns, such as conflict; living in the future; cultures and identities; jobs, money, and careers; or the environment…Planning then proceeds directly to creating activities to address the theme and related issues. There is no intermediate step in which attempts are made to identify which subject areas might contribute to the theme. (Beane, 1995, p. 619)

While Beane's definition leans on the theme driving the curriculum with little attention paid to which subject areas are contributing, Parker (2005) provides a definition that accounts for the deliberate planning of multiple disciplines to explore a theme:

> A curriculum approach that purposefully draws together knowledge, perspectives, and methods of inquiry from more than one discipline to develop a more powerful understanding of a central idea, issue, person, or event. The purpose is not to eliminate the individual disciplines but to use them in combination. (p. 452-453)

Loepp (1999) also uses the metaphor of a layered cake versus a marbled cake when discussing how different sciences might be integrated into a general science course:

> The layer cake means each of the sciences maintains an identity in a general science course while the marble cake is more problem based with the various sciences contributing to the solution of the problem. They argue that the layer cake is more of an interdisciplinary approach to curriculum because the boundaries among the disciplines are maintained. (p. 21)

Loepp (1999) also identifies three different models of integration: interdisciplinary (similar to how many U.S. public schools are set up where traditional/core subjects have set block of time, but connections are made across subjects), problem-based (a problem is at the center and multiple disciplines are necessary to reach a solution), and theme-based (where a theme directs the integration).

Despite the differences in definitions and nuances between interdisciplinary approaches and Beane's (1995) curriculum integration, what scholars do agree on is that it is a difficult endeavor to put theory into practice when it comes to curriculum

integration (Hinde, 2005; Loepp, 1999). In their study of integrated curriculum and 21st century skills, Mohr and Welker (2017) found three common aspects mentioned within curriculum integration literature: "consideration of the 'real world,' authentic connections among content areas, and negotiation of content with students" (p. 25). They also differentiate multidisciplinary approaches as still maintaining distinction among disciplines whereas interdisciplinary approaches may blur the boundaries between content areas. Exploring the literature on curriculum integration highlights the different approaches and perceptions of what it actually looks like in practice, as well as the need to ensure there is a shared understanding of what "integration" looks like in order to be able to implement it.

When it comes to curriculum integration research, the focus is often on how teachers, especially K-8 teachers more than high school and college, can effectively use it as a pedagogical tool because when done with preparation and intention, curricular integration can improve students' attitudes toward schooling, increase intellectual curiosity, enhance problem-solving skills, and lead to student achievement (Hinde, 2005; Loepp, 1999). Loepp (1999) recommends the following major aspects/supports if curriculum integration is to be successful: a shift in teacher belief system to constructivism; professional development; learning communities (with teachers and with students); guidance for teachers to become skilled facilitators of small group learning; management of experiential-oriented instructions; the use of authentic assessment strategies; supportive administration; an informed community; and systemic reform in how teachers are prepared, certified, and assessed.

Despite the benefits of and belief in the value of integration, teachers often face challenges when they attempt integration. In Margot and Kettler's (2019) systematic literature review of teachers' perception of STEM integration, they found that while secondary (grades 6-12) teachers can identify the cross-curricular attributes of STEM, they also perceive barriers to integration. Some of the common challenges they found in studies included concerns related to limited communication between subject area teachers, difficulty conceptualizing interdisciplinarity, scheduling and conflicting content standards, their own self-efficacy in STEM impacting their confidence in making connections across content, a need for professional learning opportunities, and the necessity of a quality curriculum. Margot and Kettler's (2019) study brings teachers' insights and experiences into the integration conversation and also highlights why many of Loepp's (1999) recommendations are necessary for integration to be successful.

In a time where school initiatives are often not sustained (Mohr & Welker, 2017) and many teachers are being told to "integrate" new curriculum into existing curriculum within the same structure of schooling where subjects have clear boundaries (e.g., ELA is at 8 am, Math is at 10 am, etc.) and prescribed curriculum, these challenges illustrate the complexity for teachers to actually see it to fruition (Heafner, 2018). The research and approaches of integration within the context of K-12 computer science and computational thinking education provide further insights.

K-12 CS & CT Integration

Scholars and research note the value of integrating computer science and computational thinking across and into other disciplines rather than only as a standalone

subject. This has often looked like studies exploring the design of professional development to promote CS/CT integration into other content areas and the resulting impact on teachers and/or student learning (Alqahtani et al., 2022; Coenraad et al., 2022; El-Hamamsy et al., 2021; Hubbard, 2017; Ottenbreit-Leftwich & Yadav, 2021). Mouza et al. (2018) point out that when studying computing, it is difficult to separate CS concepts from other computing-related areas like digital citizenship and computational thinking, so integrating CT or CS may (un)intentionally incorporate the other, which is why this study explores both under the umbrella of "computing."

Researchers often advocate for integrating computer science within other subject areas in order to make real-world connections, as well as other benefits, such as creating an innovative CS curriculum, appealing to girls, increasing access, being more sustainable, and expanding problem-solving (Denning et al., 2017; Miller, 2020; Ozturk et al., 2018; Sahami et al., 2013). Ozturk et al.'s (2018) PD program aimed to help elementary educators integrate CS education into their everyday lessons with projects that brought in CS skills and concepts. They found a project-based learning (PBL) approach in their PD to be impactful for teachers to envision an interdisciplinary approach of integrating CS while also identifying that teachers' concerns about standards and testing inhibited their willingness to integrate CT, so the existing standards often guided the CS project activities. Similarly, in their qualitative study with five K-12 CS teachers, Zhu and Wang (2023) identified both connecting the classroom to the real world and project-based assessment as two key teaching strategies for CS.

While some see the value of CS being both a standalone subject (e.g., Fluck et al., 2016) as well as one that is integrated, many scholars agree that computational thinking can act as an interdisciplinary field to support current standards in K-12 education (Mouza et al., 2017; Xing & Zhang, 2020) and can act as a gateway to computer science (Moudgalya et al., 2021). Since CT has been touted as a twenty-first century skill and has the potential to leverage the opportunity for all students to become interested in computing, including it in compulsory education may also help to address the disparities and inequities within the field of computer science (Yadav et al., 2018).

While many countries and some states have integrated CT/CS at the secondary level, there is also a push for integration of both at the primary level (Bocconi et al., 2016; Coenraad et al., 2022; Guo & Ottenbreit-Leftwich, 2020; Ketelhut et al., 2020; Kong et al., 2023; Ozturk et al., 2018). Estapa et al. (2018) note how elementary students are often on the receiving end of technology as they learn how to operate it but are not taught how to develop or create with it. Duncan et al. (2017) assert that integrating computing with other curriculum works especially well for primary school since there is usually only one teacher for many of the subjects, which makes it more cohesive and is easier for them to relate different ideas together. Integration at the primary school level alleviates concerns of trying to bring in a new subject into an already crowded curriculum. Duncan and colleagues (2017) believe "integrating CT with other subjects can enhance learning in these areas, and is likely to make other learning more applicable in our digital world" (p. 8).

Examples of computing integration in K-12 schools include: a physical education teacher integrates sequencing concepts from programming to create an unplugged fitness program; a math teacher uses Scratch (a free and popular programming language and online community) to teach geometry and another math teacher has students program robots to work on counting; students build video games using *Bootstrap:Algebra* to learn functions and variables in math; a language arts teachers uses ScratchJr for students to rewrite fairy tales to combine literacy, creative writing, sequencing, and computing; students create a chatbot to represent a character in a novel they are reading for language arts; students build data visualizations in social studies to support data literacy and historical inquiry (Duncan et al., 2017; Guzdial & Shreiner, 2021). While there are many examples of how to integrate CS/CT into curriculum, there are significant barriers including "access to technology, resources, time, student demographics, environmental conditions, pre-service and in-service teacher knowledge, and professional development" (Tran, 2019, p. 23).

While initiatives encouraging the teaching and/or integration of computational thinking and computer science into secondary education are not new, there have been fewer initiatives at the primary school level (Tran, 2019). The research that has been done has reinforced the importance of early exposure for elementary learners and the studies have been integral to "raising awareness in equity issues that have persisted in the evolving technology landscape for underrepresented and underexposed students, particularly females, minority populations, English learners (ELs), and children from economically disadvantaged backgrounds" (p. 5). In addition to the positive cognitive

effects found in research focusing on elementary-aged children who participate in CS/CT (e.g., improvement in mathematics, reasoning, and problem-solving skills; sustained attention and self-direction), research also highlights how CT can help increase understanding human behaviors, efficient problem-solving and error correction, and heuristic reasoning (Tran, 2019). Tran's study (2019) of 200 elementary-aged students participating in a 10-week curriculum that exposed them to CS interdisciplinary content learning in their classes found that early exposure encouraged positive attitudes about CS learning, enriched CT and problem-solving skills, and increased perseverance in solving problems, and interest/connections about CS in everyday life. Not only has CT been found to be easily integrated into many academic domains, but findings show that children have been able to learn these concepts and processes at very early grades (Tran, 2019).

Ultimately, more research is necessary to examine what CT looks like in K-12 education beyond just computer programming (Yadav et al., 2018). Tran (2019) notes that "teacher content knowledge, pedagogical practices, and instructional strategies are important factors to explore in future research" (p. 23). If computing is meant to be cross-curricular, all content teachers need training and preparation to introduce and reinforce CT/CS skills through their current curriculum and be able to use CT vocabulary to describe problems and solutions when applicable (Barr & Stephenson, 2011; Mouza et al., 2017). More exploration is needed to see how teachers transfer the skills and knowledge they develop during training(s) into their classroom practice (Bower et al., 2017). For example, Coenraad et al. (2022) also found in their study that even after PD

focused on CT integration, teachers "integrated" CT differently so it did not automatically result in an equitable experience for students. Teachers integrating computing in different ways results in different experiences for the students. Teacher preparation and their experiences of integrating computing continues to be of interest to researchers as it is necessary to achieve more equitable integration of computing into subject areas.

Teacher Preparation for CT & CS

Emerging research on teacher preparation for CS/CT teaching and integration tends to focus on either pre-service educator or in-service educator approaches. Guo (2022) shares that,

> As the push for preparing students for a CS focused workforce and promoting Computer Science Education in K-12 education proliferates, teachers are: (1) facing the policy changes and mandatory requirements of teaching CS at an early elementary school level, (2) less ready to teach CS, in terms of CS content knowledge and CS pedagogical content knowledge, and (3) expected to teach CS with other disciplines. (p. 10)

Hubbard (2017) also points to the shortage of CS teachers at both the primary and secondary level in the U.S. due to the limited pre-service teacher pathways. In their effort to understand why there is a shortage of CS teachers, the findings from Zhu and Wang's (2023) study revealed that despite the need for a foundation in the field of CS, a huge challenge for U.S. K-12 teachers is a CS knowledge gap because the potential CS teachers come from such diverse backgrounds. In response to the inadequate number of

K-12 CS teachers, alternative pathways for in-service teachers to get CS teaching endorsements and accreditation have emerged, but these pathways also make it likely that teachers can enter CS classrooms without adequate preparation (Hubbard, 2017). In response to this need for teachers to be prepared to teach/integrate computing, more research has been done to explore the ways that teachers are prepared through teacher education programs and/or participation in professional development (Kong et al., 2023; Moudgalya et al., 2021; Sadik, 2017).

Studies with pre-service educators often share the creation of a module(s) around CT/CS that is embedded into an existing course and then examines its impact on the pre-service teachers (e.g., Mouza et al., 2017; Yadav et al., 2014; Zha et al., 2020). Some of the findings of these studies of pre-service course design/integration of CT have identified the importance of going beyond just using programming to teach CT, but instead push for using discipline-based integration methods which can positively influence pre-service teachers' understanding and dispositions toward CT (Mouza et al., 2017; Yadav et al., 2014). Similar studies are done with in-service teachers and their participation in specific PD workshops that also explore the participants' resulting CT development or changes (e.g., Bower et al., 2017; Kong et al., 2023; Leonard et al., 2018). In their study of PD for pre- and in-service teachers on integrating CT into elementary science, Coenraad et al. (2022) found the levels of integration differed between exist (labeling already present CT), enhance (adding in CT components), and extend (using CT activities to support science learning). The methods used to see impacts of professional learning on these pre- and in-service educators often involve surveys,

collection of artifacts (e.g., planned curriculum documents, reflections, etc.), and observations, which occur during and/or immediately following the learning experience (i.e., Bower et al., 2017; Zha et al., 2020).

Since teachers often teach how they were taught unless a significant change occurs, El-Hamamsy et al. (2021) note that when new school wide initiatives (or pilots) are undertaken (particularly around computing integration) the following should be present: curriculum resources and assessment methods, teacher education and professional development, infrastructure and materials, adequate time, and a community. In the studies that focused on building PD to help pre- or in-service teachers prepare for integration of computing, some of the following characteristics that teachers found helpful have emerged: intentional connections of how CS/CT ideas can be integrated into core content areas, active learning and pedagogical strategies, engaging and dynamic sessions, collaborative and supportive group/environment, providing adapted content and flexibility, activities that increase teacher confidence, and long-term sustainable support (Codding et al., 2021; El-Hamamsy et al., 2021; Kong et al., 2023; Zhou et al., 2020).

As more countries and states begin to adopt computing into their school curriculums, more supports and resources will be needed for teachers as these topics will be new for many educators. Illustrative of this are the findings of Corradini et al.'s (2017) survey of 972 Italian primary teachers, which found that teachers reported that they did not feel prepared to develop CT competencies in their students or conceptualize the nuanced nature of CT. Similarly, in their review of literature, Yadav and colleagues (2018) summarized that teachers without training/support often equate CT with using

computers and lack confidence in their ability to integrate CT into their classrooms, but they identify that there is potential for teachers to shift their understanding of CT. For instance, in a study of 22 New Zealand primary school teachers in nine schools who chose to participate in professional development around CT integration, Duncan and colleagues (2017) found that teachers' confidence did not appear to correlate with time and length of teaching CT concepts, but rather related to the specific CT/CS topics being taught. Despite sometimes expressing low confidence, teachers often cited how engaged their students were in the activity, which the researchers believe will provide the intrinsic motivation to encourage them to continue adopting and integrating CT. Overall, Duncan et al. (2017) found that their pilot study showed that teachers are capable of delivering CT material in engaging ways for students when they receive adequate support, and that they can "deliver a surprising range of other curriculum topics through the computer science material, and hence the new topics take up less overall space in the curriculum than expected" (p. 10).

More research beyond surveys and self-reports is necessary to explore how computing is incorporated into teaching practices and the ways in which CT is enacted in the classroom, particularly in the U.S. (Barr & Stephenson, 2011; Bower et al., 2017; Falkner et al., 2019; Guo, 2022; Kong et al., 2023; Mannila et al., 2014; Yadav et al., 2014). Rich and Yadav's (2019) study is one of the few that studied U.S. teachers in their classrooms. In this study, elementary teachers implemented one planned CT lesson after participating in three of their professional development workshops. These lessons were captured via *Swivl* devices and most were also observed in person. Analysis of these

observations led to identification of three patterns of implementation: using CT to guide planning, using CT to structure the lesson, or presenting CT as general problem-solving strategies. Overall, there is limited research on how teachers come to understand CT and CS and how they both think about and enact embedding it in their own classrooms (Yadav et al., 2018).

Intended Curriculum

As the above sections illustrate, computer science, programming/coding, and computational thinking are being implemented in curricula around the world, including within the United States (Duncan et al., 2017; Falkner et al., 2019; Guo & Ottenbreit-Leftwich, 2020). However, just as there is no universally accepted definition of CT, there is no universal intended computing curriculum available (including required academic standards and curricula) in the United States. In their exploration of K-12 computer science curriculum standards in the U.S., Guo and Ottenbreit-Leftwich (2020) found that

> there [was] neither [a] universal method of housing CS standards into whole curriculum nor a universal CS standard across the U.S. Although some states adopted the CSTA CS framework, others developed new CS-related standards, and coordinate CS with the whole curriculum differently. Accordingly, the CS concepts covered in the CS standards were inconsistent across the states. (p. 2)

One of their findings is that many states have perpetuated the misconception of computer literacy (e.g., being able to use computers, foundational skills, etc.) and identified these skills as computer science. This finding is representative of the widely held beliefs of what computer science is, such as in Wang et al.'s study (2016) where 77.6% of 1,673

students and 74.8% of 1,013 teachers surveyed agreed that creating documents and presentations, as well as using the internet to search were a part of computer science. Knowing how CS and CT standards fit into curriculum standards is essential because it acts as a guide to teaching the content, or the enacted curriculum (Guo & Ottenbreit-Leftwich, 2020). If standards continue to perpetuate false conceptions of what CS is, then that will be reflected in the enacted curriculum. Ultimately, to make sure teachers are adequately prepared to bring computing into their classrooms and to provide equitable computing experiences, we need to understand computing teachers' enacted curriculum.

Enacted Curriculum

Nolet and McLaughlin (2000) define enacted curriculum as the operationalization of the intended curriculum (the standards or the learning goals), including the decisions teachers make in their classrooms as far as what they teach, and how. This study uses Remillard and Heck's (2014) extension of the definition which also incorporates teacher beliefs, knowledge of pedagogy and curriculum, access to resources, understanding of student needs, and local context. Since there are no universal standards for CS or even a universal definition of CT, some researchers and practitioners utilize the K-12 computer science framework, the CSTA CS Standards, and/or ISTE's definition of CT (Leonard et al., 2018) with its components and competencies to guide curriculum and lesson development (see Figure 2). During training and learning experiences, educators are exposed to conceptualizations of CS and/or CT (intended curriculum) and to different teaching strategies that they are then encouraged to bring into their own classrooms and content areas. For example, in the study by Duncan and colleagues (2017), they noted

that teachers often adapted lessons and activities for their own classroom context to account for the students' prior learning and any related materials that were being studied at the time - but that experience and outcome of adapting was not captured through observations and relied on teacher self-reports. Similarly, Yadav et al. (2018) note that while their study provided a way to investigate teachers' conceptions of CT and shifts in those ideas over the academic year, it did not capture how teachers actually translate CT integration into their instruction, or the enacted curriculum. Ozturk et al. (2018) used fieldnotes from PD meetings and teacher reflections to better understand how elementary teachers experienced using project-based learning to integrate CS. To align both the intended and enacted curriculum, Ozturk et al. identified the need for more consideration of influences on the enacted curriculum, such as timing and school schedules, resources, infrastructure challenges, and teachers' pedagogical and content knowledge.

This dissertation study addresses the gaps in literature by exploring how teachers pedagogically approach bringing in the intended curriculum of computing (enacted curriculum). It is also helpful in identifying problematic areas of intended curriculum, noticing key supports to guide future resource development, and creating more alignment between intended and enacted curriculum (Falkner et al., 2019).

Minnesota's Computing Education Context

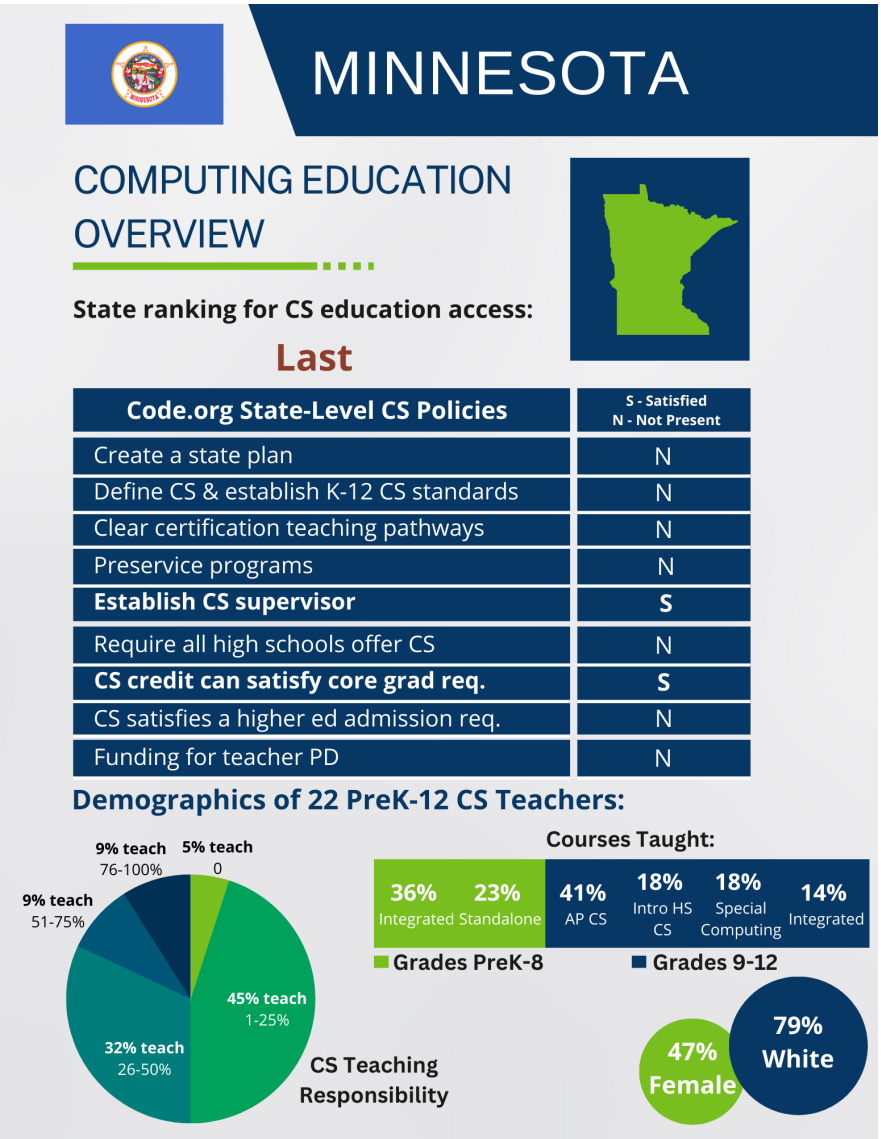


Figure 3. An overview of computing education in Minnesota. The overview highlights how the state meets Code.org's nine state-level CS policies (Code.org et al., 2022) and Koshy et al.'s (2022) summary of 22 MN PreK-12 CS teachers demographics.

Minnesota, located within the traditional homelands of the Anishinaabe and Dakota tribes, is found in the upper Midwest of the United States and has a population of over 5.7 million people (U.S. Census Bureau, 2021). Despite having a reputation for some of the best quality education in the United States (CBS News Minnesota, 2020),

Minnesota currently ranks last in the nation for access to CS education as only 21% of its high schools offer computer science and the state has only two of the nine state-level CS policies recommended by Code.org: establishing a state CS supervisor and allowing CS to satisfy a graduation requirement (Code.org et al., 2022). Currently CS education opportunities in Minnesota public schools depend on initiatives from the individual teacher, school, and/or district. Beginning in 2018, the MN Department of Education began integrating CS into the K-12 Minnesota Academic Standards but left it up to each Local Educational Agency to make curriculum decisions and course pathways, so they are dependent on individual districts and schools (MN Department of Education, n.d.). This means Minnesota has no state-level initiatives or policies regarding standalone K-12 CS standards for students, funding for CS professional development, CS teacher licensure certifications, and no statewide CS education plan (Code.org et al., 2022; Office of the Legislative Auditor, 2016).

Despite ranking last in the US for access to CS, the state is making progress in supporting more opportunities for computer science education. Minnesota joined the national Expanding Computing Education Pathways Alliance (ECEP) (https://ecepalliance.org) alongside 23 other states and U.S. territories in the fall of 2018. Minnesota's ECEP alliance chapter, CSforAll-MN, works to help and advance the vision for equitable K-12 CS education in Minnesota with a focus on policy, broadening participation, and professional development (https://csforallmn.org). They are currently focused on promoting state policy, particularly around advancing a foundational state

blueprint for computer science education (CSforAllMN, n.d.). CSforAll-MN identified that,

> There are a number of factors and realities that impact the state's ability to offer equitable public computer science education that are connected to histories of persistent educational achievement gaps for Black, Hispanic, and Indigenous students; growing immigrant communities (half a million in 2020); differences in socioeconomic statuses, and economic and cultural differences between urban and rural areas in the state. These contextual realities are not the only reasons why Minnesota's CS education offerings are limited, but they must be considered as reasons why opportunities and access for all K-16 students to computer science is important. (CSforAllMN, 2020b)

Until there is state-wide policy or guidance around CS education, it will remain up to individual teachers, schools, and districts to bring opportunities to students which ultimately leads to perpetuating the inequities already visible in computing. In the meantime, much can still be learned from those Minnesota educators currently trying to integrate computing.

In addition to not having CS course pathways for students, Minnesota does not have K-12 teacher licensure specific to CS (MN Department of Education, n.d.) - in fact only two of the thirty-two higher education institutions in the state prepare teacher candidates for an initial business education license, thus allowing them to teach CS (CSforAllMN, 2020c). Instead authorization is given to certain licenses (e.g., a career and technical education (CTE) license) to teach CS, providing pathways for teachers to

demonstrate their competency and thus resulting in K-12 CS teachers with a wide array of backgrounds and experiences. In their national survey of PreK-12 CS teachers, Koshy et al. (2022) had twenty-two teachers from Minnesota respond. Of those twenty-two Minnesota teachers, 47% were female and 78% identified as white. Forty-one percent of the teachers had no CS classes in college, 50% had some CS classes, and 10% identified having either a CS minor or CS major. Forty-one percent of the teachers taught advanced placement (AP) CS courses in grades 9-12 compared to 14% who identified teaching integrated CS courses with grades 9-12. Comparatively, 36% of the courses taught with grades prek-8 were integrated and 23% were standalone elementary CS courses. Despite the fact that overall, 73% of the teachers were considered late career, when it came to teaching CS, 50% considered themselves mid-career and 32% considered themselves early career. The responding MN CS teachers were least confident in using culturally relevant pedagogy to support student learning (57%). This is just one snapshot of the CS teachers who responded within the state of Minnesota.

Based on the current CS landscape, Minnesota is unique in that teachers who are integrating and/or teaching CS/CT have not had a shared foundational knowledge of CS (either in the form of CS standards or teacher licensure program) but instead may be representative of a variety of experiences and backgrounds: perhaps they are classroom teachers trying to integrate it because of their own interest/investment or as a result of trying to interpret the MN K-12 integrated CS and academic standards, or they had industry backgrounds and made the switch to education, and/or they may be content-specific teachers who also qualify to teach CS. As Minnesota works to advance equitable

CS opportunities for all students, seeing what is happening in different classroom contexts across the state where teachers have had some training experience can help illuminate their enacted curriculum including what is working when preparing teachers to teach or integrate CS/CT, as well as what challenges await.

Conceptual Framework

In combination with computational thinking, the technological pedagogical content knowledge (TPACK) framework (Koehler et al., 2013) (see Figure 4) guided this study as I explored how teachers enact TPACK related to computational thinking in their classrooms. TPACK builds upon Shulman's (1986) construct of pedagogical content knowledge (PCK) to incorporate technology knowledge (TK). The TPACK framework honors the complexity and dynamic work of teaching as it allows for the exploration of educators' content knowledge (CK) (i.e., knowledge of the subject matter), pedagogical knowledge (PK) (i.e., methods of and approaches to teaching), technological knowledge (TK) (i.e., technologies and associated terminology), and their relationships with each other – all situated within a specific context. The interactions amongst these constructs include pedagogical content knowledge (PCK) (i.e., methods and strategies for representing/teaching content), technological content knowledge (TCK) (i.e., using appropriate technology/terminology to create/interact with the content), technological pedagogical knowledge (TPK) (i.e., understanding of how technologies can be used in teaching), and then technological pedagogical content knowledge (TPACK) (i.e., all of the domains interacting to support effective use of technology aligned with the content area).

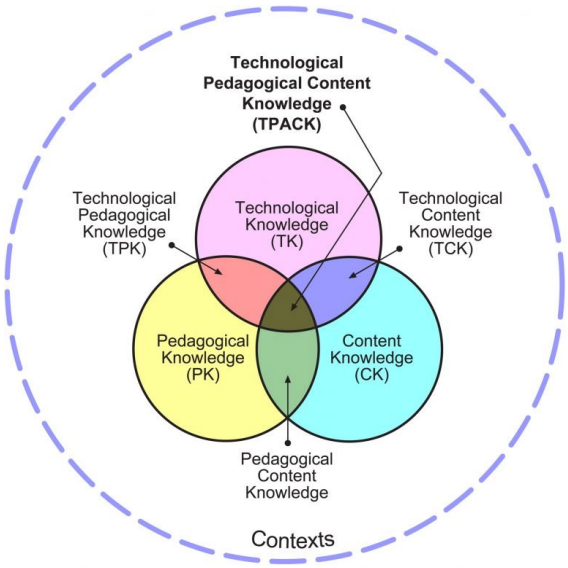


Figure 4. The TPACK Framework (Kurt, 2018).

TPACK is complex, much like teaching, and identifies the cognitive knowledge teachers need to have to integrate technology (Koehler et al., 2013). In addition, the framework is popular in education research as it has been used in over 600 journal articles (Willermark, 2018), including as a tool to identify and name how an educator teaches by providing a lens to facilitate and/or analyze pre- and in-service teachers' approaches to and understandings of technology integration (Eichelberger & Leong, 2019; Stoilescu, 2015; Voogt et al., 2013). Recent work has even used TPACK as a framework for designing CS professional development around culturally responsive pedagogy (Codding et al., 2021) - the framework allowed researchers to focus on how the PD could strengthen teachers' content knowledge (CK) and pedagogical approaches to instruction (PK & PCK).

A major criticism of the framework is finding the definitions of the different cognitive knowledge domains to be insufficient making them harder to identify *in*

practice (Willermark, 2018). Yet this study's purpose is not to identify how competent a teacher is in each knowledge base or to rate a teacher's TPACK, but rather my dissertation used TPACK as a tool to observe, identify, and analyze the complex task of how teachers integrate CT/CS in the classroom. There is a need to go into classrooms and observe what teachers are actually doing and how their cognitive conceptualizations of CT are being enacted into their classroom (Yadav et al., 2018). I used TPACK to help make visible these expressions of knowledge (e.g., pedagogical strategies, lesson plans illuminating the teachers' understanding of content and CT's role, etc.), the enacted curriculum, by using TPACK combined with CT; TPACK-CT (see Figure 5) provides a holistic way to articulate, identify, and organize what knowledge(s) teachers are pulling from in their effort to integrate CT into their classrooms.

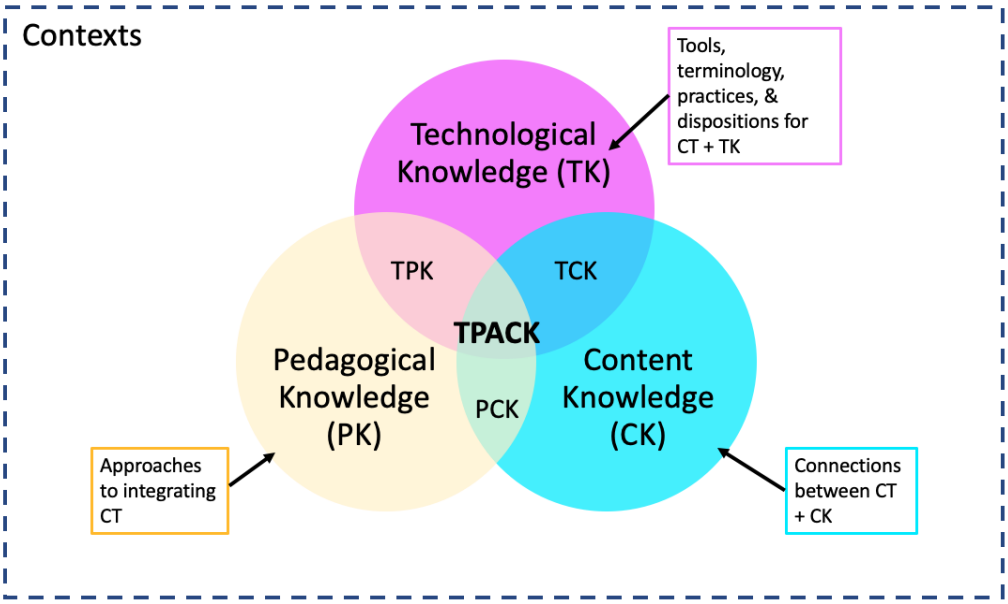


Figure 5. A visualization for the TPACK-CT framework.

My work also builds off the few studies that use TPACK as a tool for analyzing pre-service teachers' conceptualization and teaching of computational thinking,

reframing it as TPACK-CT (Angeli et al., 2016; Mouza et al., 2017). Angeli and colleagues (2016) provided a conceptualization of TPACK with a focus on the knowledge (CK) teachers need to teach standalone computing courses, emphasizing the content knowledge (i.e., abstraction, decomposition, debugging, etc.). Mouza and colleagues (2017) expanded upon this work by examining what *all* K-8 pre-service teachers need to know if they are going to integrate/teach CT. Their expansion includes looking for the computing tools, terminology, practices, and dispositions for cultivating understanding of CT (TK-CT), making connections between CT and existing content (CK-CT), as well as the pedagogical methods used to make these connections (PCK-CT). Rather than focusing or measuring teachers' knowledge related to their content/expertise, this study utilized their conceptualization of TPACK-CT with a focus on in-service K-12 teachers' understanding of "how CT-related concepts, computing tools, and practices (TK) can be combined with disciplinary content (CK) and pedagogical strategies (PK)...in specific contexts" (Mouza et al., 2017, p. 63).

This framework fits well within the constructivist paradigm as this study did not pursue one single truth or "right way" to approach computational thinking. Rather, this framework served as a holistic lens for answering the research questions as it was used as a methodological tool to analyze the data collected. It also aided in further understanding the complexity of integrating computational thinking into existing curriculum, including the role of technology (Koehler et al., 2013). I used the TPACK-CT framework as a tool to interpret teachers' enactment of computing in the classroom and to provide ways for teachers to articulate their approach to the different domains of this framework.

Ultimately, the TPACK-CT framework served this study as a lens and guide for identifying and interpreting the nuanced job of teaching and integrating computing in the classroom.

Summary

This literature review sheds light on how computing education, particularly computational thinking and computer science, is being advanced worldwide with an emphasis on its emergence and implementation in the context of the United States, often as a response to the lack of diversity in computing. Throughout the U.S. there is not only a push to integrate CS/CT into core curricula starting in elementary school in order to ensure all students will have access to computing, but there are different approaches to implementing state standards for CS, a variety of pathways for teachers to become CS teachers, and different professional development opportunities emerging in response to the gaps in computing knowledge. While there are studies and research being done on the success of different computing development experiences for pre- and in-service teachers, most rely on methods that do not capture the enacted curriculum of what is actually happening in the classroom after the PD experience concludes.

Minnesota is reflective of many of the challenges to integrating and teaching computing as it does not have standalone CS standards nor CS licensure pathways nor teacher education programs. Instead, Minnesota has begun embedding computing standards into other core content standards and relies on the individual districts and schools to interpret and implement them. As a state that currently ranks last in terms of high schools offering CS, but who has shown a commitment to making progress to

support and expand opportunities for computing education, it is important to understand the lived realities of the teachers who have begun doing this work. By using the TPACK-CT framework as a lens and a multicase study approach, I shed light on the enacted curriculum of three teachers who are working to integrate CT/CS into their curricula and provide further insight into the supports and challenges they have encountered as Minnesota works to prepare teachers to integrate computing.

CHAPTER 3: METHODOLOGY

The purpose of this study was to observe teachers' enacted curriculum of integrating computational thinking (CT) and computer science (CS) into their classroom; the focus was on identifying what actually happens in the classroom beyond participation in professional learning (e.g., a workshop, coursework in their degree program, etc.). Utilizing the technological pedagogical content knowledge (TPACK) framework (Koehler et al., 2013) with the incorporation of computational thinking (TPACK-CT), this comparative multicase study explored how educators integrated CT/CS into content areas, or their enacted curriculum. It was approved by the University of Minnesota's Institutional Review Board (see Appendix A). The following main questions and supporting questions guided this research study:

- What does teachers' TPACK-CT look like when enacted in the classroom? *[technological pedagogical and content knowledge related to CT, TPACK-CT]*
    - How do teachers conceptualize computational thinking knowledge in their classroom? *[content knowledge, CK-CT]*
    - What pedagogies are being enacted in the classroom when teaching CT/CS? *[pedagogical knowledge, PK-CT]*
    - What technologies, if any, do in-service teachers utilize for teaching CT/CS? *[technological knowledge, TK-CT]*
- How do teachers' TPACK-CT compare and contrast across cases?

This chapter begins with an overview of comparative multicase studies, details on state and participant selection and recruitment, and an overview of participants - including

myself as I share my role as the researcher. Next, I outline my data collection, including how it aligns with the research questions, when it occurred, and what was collected. Last, I share my approach to data analysis and validity strategies.

Comparative Multicase Studies

Comparative multicase studies (Merriam & Tisdell, 2016) explore several cases through detailed data collection of multiple sources of information; they can result in a "more compelling" interpretation than a single case study (Merriam & Tisdell, 2016, p. 40). Using a survey, observations, interviews, reflections, and curriculum artifacts, this comparative multicase design allowed for better understanding and interpretation of teachers' TPACK-CT by constructing individual cases bounded by each in-service teacher (3 cases). When working with multiple cases, each case is first analyzed and the findings are presented as an individual case, or "portrait" (Merriam & Tisdell, 2016): "Conveying an understanding of the case is the paramount consideration in analyzing the data" (p. 233). The cross-case analysis, or synthesis, of the three cases occurs after each of the within-case analyses and is presented to see what themes emerge in answering the research questions and to notice similarities and differences of teachers' TPACK-CT across their different classrooms.

As a researcher with constructivism being a large part of my orienting epistemology, I chose this qualitative design because this study is ultimately concerned with "discovery, insight, and understanding from the perspectives of those being studied" (Merriam & Tisdell, 2016, p. 1) rather than determining a "right way" or the "best method" to integrate computing into curriculum and classrooms. In combination with the

TPACK-CT lens, this methodology not only allowed me to construct an understanding of what is happening in individual classrooms, but it also honored the complexity of integrating CT/CS, something that is negotiated within the classroom context. Through this qualitative comparative multicase study, I hoped to better understand and share what enacting TPACK-CT looks like in multiple contexts, and to understand what lessons around integrating computing can be learned within and across each context.

State and Participant Selection

As policies and curriculum related to computational thinking and computer science education vary so much across the United States, one state was chosen to recruit participants from: Minnesota. I was living in Minnesota at the time, and this ensured that I, the researcher, could account for a deeper, more nuanced understanding of the context of the state and allowed for a greater likelihood of being able to visit the classrooms given the rapid changes occurring in schools and with travel as a result of the COVID-19 pandemic. Since there is not a computer science licensure endorsement in Minnesota (CSforAllMN, 2020c) nor a system tracking teachers' integration of CT or CS, participants were recruited by tapping into a network of known computing trainers and higher education institutions offering CS/CT classes for educators within Minnesota and asking them to share a call for participants.

The initial strategy was to find two to three teacher participants from different grade bands (ideally grades 3-5). Parameters for selecting participants were that they were current teachers, intentionally trying to integrate CT or CS into their classroom, have had some type of professional development experience related to CT/CS,

demonstrated understanding of CT, and that there was diversity in locations and students served. The ideal parameters for selecting participants originally included grade level (grades 3-5) and professional learning experience with CT within the last 2 years. However, because of the limited number and availability of respondents given the year and its challenges (particularly COVID-19), the number of teachers was reimagined to be three or more teachers (discussed more below) and parameters were expanded to include any teacher, including specialists, working with primary or secondary grade levels. Since the goal of this study was to learn more about teachers' enacted curriculum, or TPACK, when integrating CT within the classroom, and not to examine the effectiveness of the professional learning experience nor to determine the best way to integrate computing, the parameters (exact number of participants, grade level, and years since professional development) were flexible while still preserving the original intent of the study and ability to address the research questions. Participants were recruited that have a variety of experiences and taught different grade levels – resulting in richer case construction.

Initially trying to recruit elementary teachers was a strategic decision for the following reasons: elementary curriculum in theory easily allows for the integration of computing into many content areas since one teacher teaches multiple subjects (e.g., math, science, language arts, etc.) and there is a need for more research on computing integration in elementary education. However, expanding the study to include secondary teachers still allowed for the investigation of integration of CT/CS and expanded the possibilities of gaining insights from examining the possibilities (or lack of) for integrating computing at different levels (e.g., noticing if teachers in 5th grade used more

advanced CT concepts than those in 10th grade ELA, what pedagogical decisions were the same/different across grade levels to introduce similar concepts, etc.) and in different contexts (e.g., does a STEM specialist have different resources available than a fourth grade teacher). Funding was also used from the PEO Scholars award to incentivize participants by providing compensation of a $175 Amazon gift card or cash to each participant - participants also received surprise coffee gift cards throughout the study with this award.

Recruitment

Since there is no statewide database for Minnesota of teachers currently bringing CS/CT into their classrooms, I reached out to well-known organizations and districts in Minnesota that I have knowledge of and/or had partnerships with and that I know include computing in their curriculum/PD to ask them to forward my recruitment email to their past participants/cohorts. Emails asking contacts to forward on a recruitment email to potential participants were sent to contacts associated with Code Savvy, St. Catherine's University, College of St. Scholastica, the Minnesota chapter of Computer Science Teachers Association (CSTA), and additional University of Minnesota researchers and staff working with teachers throughout the state. The recruitment email (Appendix B) provided a brief overview of the study, including what participating in it would look like, compensation details, and then directed respondents to indicate interest by filling out a short Qualtrics form (Appendix C) to capture their location, computing training experience(s), grade levels and content areas, an example of how they have integrated CT into a lesson(s), and availability in order to identify research participants.

Since the recruitment for this study took place in the summer of 2021, the impacts of the COVID-19 pandemic on educators were ongoing and significant. Many teachers were still recovering from the 2020-2021 school year when their schools and classrooms went remote, and there was much uncertainty about what the 2021-2022 school year would look like (online vs. in-person vs. hybrid). This was not only a factor in the number of teachers willing to commit themselves, their time, and their energy to participating in research the following year (or trying to imagine how they would do computing after the year they just experienced), but it also helps account for the sudden changes in teachers' roles/schools as the 2021-2022 school year became more defined and they no longer wanted to/were able to participate.

Eleven teachers indicated interest in participating while the interest form was open from May to September in 2021, with most recruitment happening during May-July. Potential participants were color-coded based on context(s) and grade levels to see where there was overlap - at first glance, there was the potential to have two participants per context/grade level. After reaching out to the pool of participants via email to confirm their interest and answer any questions, the pool was narrowed down from eleven to three as some teachers did not reply, communicated that their focus was more on using technology rather than intentional integration of CT/CS skills, were moving to a new school or using a new curriculum and no longer had the capacity to participate, and some opted out as they felt they were not a good fit after corresponding (see Table 1). The responses to the interest form also revealed a preview of the different contexts of teachers in Minnesota working to integrate computing, as well as the wide array of perceptions of

what integrating CS and CT looked like as some teachers mentioned projects that involved using technology (e.g., slideshow software) rather than focusing on CT or CS concepts - although it was always planned to have more clarification during the survey when participants agreed to the study. Districts of the three respondents who retained interest and wanted to participate were contacted in order to ensure support for them to participate and to obtain permission for me to visit their classrooms.

Table 1

Overview of Respondents to Recruitment Survey

Potential Participant ID	Grade Level(s)	Content Area	Lesson Example of integrating CS/CT	Exclusion Reason
1: Mrs. Bee	K-5	Elementary STEM	Design and creating engineering projects using different resources (Scratch, code.org, iPad apps, etc.)	N/A; INCLUDED
2	K-5	Technology & Makerspace Specialist	Use Scratch to make stories, shapes, and life cycles, etc.	Opted out due to a move to a new school.
3	9-12	Math & CS	Sprinkle CS in Algebra courses.	Opted out while getting district approval.
4: Mr. Ozo	4	Core Subjects	Math lessons with CS concepts & makerspace materials during morning meetings.	N/A; INCLUDED
5	2	Core Subjects	Code.org curriculum & incorporate ISTE standards.	Opted out while getting district approval.

Potential Participant ID	Grade Level(s)	Content Area	Lesson Example of integrating CS/CT	Exclusion Reason
6	9-12	Science	CT principles in physical science to balance chemical equations (including abstraction to understand interlocking blocks and algorithms to create a balanced chemical equation).	Did not reply to follow-up emails
7: Mr. Pi	9-12	CS	History, ethics, music, art, English (storytelling), etc.	N/A; INCLUDED
8	K-4	Spanish & Technology	Direct CS instruction & using robots during Spanish lessons to practice vocab.	Did not reply to follow-up email after answering questions
9	6-12	Drama & Tech	Maintaining and repairing tech equipment for the district, technology integrated in shows & students trained to manage it.	Opted out + interpretation of CS was not a good fit.
10	6	CS & Tech integration (6-8)	Standalone CS and integrated science unit on climate simulation, as well as an arduino unit for electricity learning.	Did not respond to email.
11	5-8	Media Tech	Using Google Slides and a voice recording to create an eBook.	Considered creating a slideshow to be integrating CS/CT.

Participants

Three participants were chosen based on their responses to the initial interest form, their willingness to participate, their schools' support, and their diverse experiences

(see Table 2). They each taught in different areas, had previously participated in different professional development (PD) and learning experiences, had different roles and grade levels they worked with, and were able to confidently explain CT/CS.

Table 2

Participants

Name*	Role	Years of Teaching Experience	Years of Integrating CT/CS	Frequency of teaching CT/CS	Last CT/CS-related PD Experience
Mrs. Bee	STEM specialist for grades K-5	27	6	2-3x/week, 16-30 mins	2017-2018 + workshops when possible
Mr. Ozo	Grade 4 teacher	17	4	Every day, 15 mins or less	2017-2018
Mr. Pi	IT & CS teacher for grades 9-12	9	9	Every day, 46+ mins	Prior year (2020-2021)

**Names are pseudonyms*

While the original goal had been to find multiple teachers in the same grade band, I determined that the research questions could still be answered by focusing on a smaller number of participants and their variety of experiences would lend even more depth and nuance to understanding what is happening in Minnesota classrooms.

My Researcher Kaleidoscope

In qualitative research, "the researcher is the primary instrument for data collection and analysis" (Merriam & Tisdell, 2016, p. 16), which is intentional as humans are both "responsive and adaptive," making them ideal for the collection and analysis of qualitative data. While there are advantages to being the human instrument in research

(e.g., providing insight into nonverbal cues, immediate processing of information, ability to clarify and summarize, etc.), it also means that I bring my own shortcomings and biases to the study (Merriam & Tisdell, 2016). I termed this my "research kaleidoscope" to capture how my unique positionalities, experiences, and histories (i.e., the tiny objects at the bottom of the kaleidoscope) impact my view of the world around me as they are reflected outward (see Figure 6). The resulting view is changing and no two people will see the exact same perspective when viewing the world around them. In addition, I like that there are mirrors within the kaleidoscope as they are an important reminder for reflection and reflexivity (and the recursive nature of it). So rather than aiming to be as objective as possible by attempting to ignore both the kaleidoscope I bring to this research or the fact that I exist in the same social processes/world I am studying (Holmes, 2020), I spent time articulating my experiences/histories/positionalities/identities, holding them up beside my conceptual framework, and grappling with the patterns they reflect on (or their role in) shaping my data collection and analysis processes.

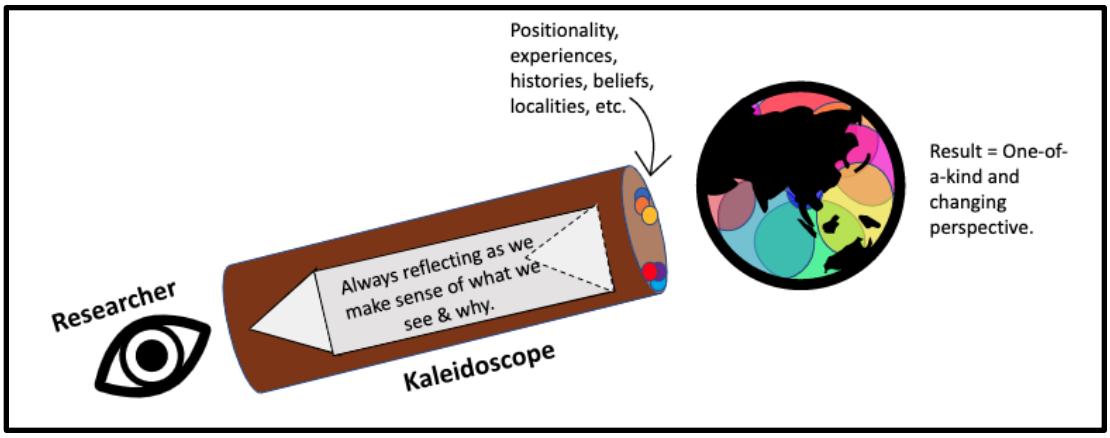


Figure 6. An illustrated conceptualization of a researcher kaleidoscope. The simplified kaleidoscope symbolizes how I conceptualize that my positionality, experiences, histories, beliefs, etc. all impact the way I see the world - and that this perspective is unique to me and it changes. The mirrors within the kaleidoscope serve as a reminder to

keep reflecting on how and why I make sense of my research, as well as capturing the recursive nature of reflection and reflexivity.

First, I recognized that as a human and researcher there was no way I would be able to separate myself from this research as I bring my whole self to the work - an ever-evolving identity formed by my histories, beliefs, values, experiences, and localities. In order to explore the ways my humanness, or kaleidoscope, would impact the research, I engaged in reflective and reflexive processes of locating the intersections of myself, participants, the world, and literature prior to and throughout this study. Sandelowski and Barroso (2002) define reflexivity:

> Reflexivity is a hallmark of excellent qualitative research and it entails the ability and willingness of researchers to acknowledge and take account of the many ways they themselves influence research findings and thus what comes to be accepted as knowledge. Reflexivity implies the ability to reflect inward toward oneself as an inquirer; outward to the cultural, historical, linguistic, political, and other forces that shape everything about inquiry; and, in between researcher and participant to the social interaction they share. (p. 216)

Prior to engaging in my research activities, I spent a lot of time reflecting on my experiences, values, and beliefs through writing to gain a better understanding of myself, and to provide myself with an articulated reminder of these influences during the research process and analysis. While I made every effort from the beginning to be painfully transparent with myself, I also acknowledge that "aspects of the self can be missed, not known, or deliberately hidden" (Holmes, 2020, p. 4). Yet through writing, struggle, and reflective (voice) memos, I still tried to account for as much as I was aware of by

articulating, returning to, and evolving my researcher kaleidoscope, including my place in the contexts involved, and my potential influences on the research.

I approach research from a constructivist paradigm where I believe reality is socially constructed and there is no single, observable reality, rather there are multiple interpretations co-existing. I seek to understand the world in which we live today and believe that interactions, culture, and history shape the meanings of our experiences (Creswell, 2013). With the privilege I have experienced and bring to situations as a white woman who is often relationally motivated, I also know that I usually tend to desire understanding via stories and experiences before taking action/pursuing change. I also identify as a pragmatist and believe that the question should drive the methods used and that more than one method can be used - real-world practice and applications drive my research inquiries, interest, and interpretation (Creswell & Clark, 2017). These approaches to research mean that because of the questions I am asking in this research, I also took an *emic* or "insider" approach in order to:

> understand situations in their uniqueness as part of a particular context and the interactions there. This understanding is an end in itself, so that it is not attempting to predict what may happen in the future necessarily, but to understand the nature of that setting - what it means for participants to be in that setting, what their lives are like, what's going on for them, what their meanings are, what the world looks like in that a particular setting - and in the analysis to be able to communicate that faithfully to others who are interested in that setting…the analysis strives for depth of understanding. (Patton, 1985, p.1)

This also means that my methodological approach for this study was inductive - I wanted to build upon participants' views and experiences to find themes rather than (dis)prove my own hypotheses or existing theories.

As a student, educator, and researcher I have seen the joy and "ah-ha" moments that technology and computing can bring to learning – particularly when student interest, choice, and identities are centered. These experiences have made me a believer in the power and possibilities of integrated technology in education while also cautious of how it is often used and touted in education spaces. Yet, I have also experienced firsthand how not having connections or representation in subjects and careers can color my view of who I am (i.e., not a math person) and what I am good at (i.e., language arts), as well as the negative impacts of a lack of exposure to computer science (e.g., living in a rural area with only a typing class) or not finding a sense of belonging in male-dominated spaces/careers (e.g., seeing women struggle in CS fields at a technical college where I was an advisor). All of these experiences combined with my career as an educator, the (un)learnings I have experienced as an educator and participant in life, and my initial orientation to research all contributed to igniting my initial passion for this study. This passion and excitement, as well as the desire to collaborate, had to be tempered at times during this study - particularly when in the classroom and during interviews.

I tried to set my teacher identity aside when I was in the classroom. While I would have loved to collaborate with my participants and share ideas, I went in with the mentality of a grateful observer and learner. My prior research had been with students so this time I intentionally worked to focus on what the teacher was doing rather than

capture the students' experiences. While I at times jotted down notes about the whole environment, I often returned to my observation protocol guide (see Appendix D) to make sure I was paying attention to the focus of my study – particularly the focus on the teacher's use of technology related to computing, CT/CS content knowledge, and the pedagogical strategies they employed. Prior to interviews, I spent a lot of time going over what participants had previously shared, reading literature, and grounding myself (again and again) in my research questions to develop questions unique to each participant, and to help dampen, as needed, my own interests as an educator/student with a desire to connect. These pre-planning strategies and observation protocols (see Appendix D) provided me both boundaries and conscious reminders of my role in this study. I also ensured that my own thoughts during interviews were kept to a minimum and always led with participants sharing their feelings/beliefs first. In addition, during cleaning of data and analysis, I made note of my own contributions to better understand and make visible my influence in that moment.

I was also cognizant of and journaled about the preconceived ideas of what was happening related to CS/CT in Minnesota based on my conversations and experiences with teachers that I interacted with in other research partnerships and based on my involvement as a steering committee member of CSforAll-MN. This is why it was important to me to use organizations for recruitment (rather than an email from me to students I had taught or teachers I had worked with in the capacity of a researcher for the University) and to have more than one participant. I also did not have any face-to-face meetings with potential participants prior to their completion of the survey in hopes of

being as little of an influence on their responses as possible. This is also something I grappled with in interpreting the findings as I was sometimes tempted to draw upon those experiences as I felt like there was a missing context that I had awareness of, so I paused frequently to engage in conversations with myself about the purpose of this study and what the data alone was revealing. I continually reminded myself that the purpose of my research was not to capture every situation nor to reveal the entire, comprehensive landscape of CS in Minnesota.

Despite my efforts to reach a wide variety of teachers, I did have a participant who had been a student of mine two years before this study in an online graduate class focused on technology tools for educators, but had since graduated. With this participant, I really emphasized that there were no right or wrong answers as I did with all of my participants, and echoed that this was about their experiences and realities - I also did not bring up the course. There were other connections I had with each participant (knowledge of the same people, similar identities, etc.), and so I tried to account for this by being aware of how I positioned myself in their classrooms (in the back, sitting, etc.), the intentional word choice in our interactions, and in the questions I asked, and by continually reflecting on our conversations. While I was in the classroom, I also purposefully interacted with the students as that seemed to put the teachers more at ease/help them feel less watched. Each teacher chose to introduce me and I let them decide how to introduce me (often as a visiting teacher or researcher), and I emphasized that I am just there to hang out and that there was no pressure to answer any questions that I ask - the participants were also told this at the start of every interview. When

students asked me questions about what they were doing/class activities, I referred them to their teacher – framing the teacher as the expert and not myself.

I conducted, recorded, and had all interviews transcribed via Temi (www.temi.com) (first interviews) and Zoom automated transcriptions (https://zoom.us) (second interviews) that I then proofread/updated and added in some notes on tones/gestures. I ensured I had district support in accordance with IRB, and in my follow-up with those interested in the study, I deliberately never shared my thoughts on their CT/CS activities and instead asked follow-up questions as I wanted to hear and learn more and avoid framing myself as an expert on computing. I continued to keep informal (voice) memos and notes throughout each stage of data collection and analysis to give myself an outlet where I could share my reflections, and continued this process during analysis and writing as I reflected on and wrestled with myself via these notes to see how they were mediating my research. Writing and data analysis were recursive and involved turning back to the data over and over again to see both what the participants said/revealed and how I had positioned myself in that situation. I also paused at times to revisit literature, to journal through what I was struggling with, and/or to remain conscious of my own beliefs and understandings.

By being the human instrument for this study, I believe that my humanness contributes to making this study a "distinctive contribution, one that results from the unique configuration of [my] personal qualities joined to the data [I] have collected" (Peshkin, 1988, p. 18). Being aware of my experiences and my shared connections/experiences (or my researcher kaleidoscope) gave me a nuanced

understanding and insight into what questions to ask and what to pay attention to in the classroom – I think it positioned me as a good fit for being the human instrument of this study. I also acknowledge that while making every effort to account for and mediate my positionality throughout this journey and study, it is ultimately "inseparable from the research findings" and an integral part of becoming a (better) researcher (Holmes, 2020).

Data Collection

To explore what teachers' TPACK-CT looks like when enacted in the classroom, data sources (observations, interviews, artifacts, reflections, and a survey) were aligned with each research question (see Table 3). The first main research question was answered through a culmination of all of the methods used to answer the sub questions, and the second research question was answered through the cross-case analysis. Data collection took place during the 2021-2022 school year (see Table 4).

Table 3

Alignment of Research Questions and Data Sources

Main Research Question	Sub Research Questions	Data Source(s)
What does teachers' TPACK related to computational thinking look like when enacted in the classroom? *[TPACK-CT]*	How do teachers conceptualize computational thinking knowledge in their classroom? *[CK-CT]*	• Survey (Appendix E) - items developed based on Yadav et al.'s (2014) Computing Attitudes Survey & Mouza et al.'s (2017) & Leonard et al.'s (2018) adaption - Appendix F • Interview(s) - Appendix G • Observation(s) - Appendix D • Curriculum artifacts - *secondary*
	What pedagogies are being enacted in the classroom when teaching CT/CS? *[PK-CT]*	• Interview(s) - Appendix G • Observation(s) - Appendix D • Teacher reflection - Appendix H • Curriculum artifacts - *secondary*

	What technologies, if any, do in-service teachers utilize for teaching CT/CS? *[TK-CT]*	• Interview(s) - Appendix G • Observation(s) - Appendix D • Teacher reflection - Appendix H • Curriculum artifacts - *secondary*
How do teachers' TPACK related to computational thinking compare and contrast across cases?		• All data sources mentioned above • Individual case analyses/portraits were used to illuminate similarities and differences across cases (teachers)

Table 4

Timeline of Study and Data Collection

Pre-Data Collection	February 2021	- Thesis proposal meeting & approval
	March - May 2021	- Finalized IRB, recruitment/consent emails & forms
	May - September 2021	- Recruitment emails went out - Identified potential pool of participants
	August - September 2021	- Enrolled 3 participants & got school approval; updated IRB
Data Collection (& Ongoing Analysis)	September - October 2021	- Surveys sent and completed - 1st analysis of surveys - Completed 1st interview with each participant
	November 2021 - April 2022	- Observations completed - Teacher reflections completed after each visit - 1st analysis of all data collected to prepare for 2nd interview
	April - May 2022	- Completed 2nd interview

Analysis	September 2021 - May 2022	- Case study database is created & maintained throughout study (Merriam & Tisdell, 2016) - Interviews transcribed - Constant analysis of data to prepare for 2nd interviews and observations - Completed initial coding of surveys, 1st interview, teacher observations and reflections, and artifacts
	May - December 2022	- Descriptive analysis of surveys (individual and then compared) - Continued 2nd round of coding of data + coding 2nd interviews - Findings for each participant categorized & summarized - Case construction begins
	December 2022 - February 2023	- Case portraits completed - Member checks of case portraits - Word clouds and counts are created as part of across case analysis (see Appendix I) - Cross-case coding table created - Thematic analysis / findings uncovered - Cross-case analysis written
Writing	February, 2021- March, 2023	

A survey, based on both the Teacher Beliefs about Coding and Computational Thinking (TBaCCT) survey (Rich et al., 2020) and Yadav et al.'s (2014) original

Computing Attitudes Survey, as well as recent adaptations (Leonard et al., 2018; Mouza et al., 2017) (see Appendices E and F), was completed by participants at the onset of the study. Since the purpose of the survey was to capture teachers' content knowledge around CT and to prepare for the first interview rather than performing a statistical analysis for validity, some questions from the sample surveys were combined, adapted, or ignored. A cognitive interview protocol (Willis, 2004) was used to pilot the study with two educators and one engineer to note any potential issues or changes that need to be made prior to the survey being shared with participants. Changes were made to clean up grammar, remove unnecessary negatives, and to clarify meanings where multiple interpretations could confuse participants. The survey collected basic demographic and context information and teachers' understanding of and disposition toward CT. The survey used Likert-scale items to inquire about the teacher's working knowledge of CT (i.e., definition), comfort level of teaching computing, interest in computing, integration in the classroom, future use of CT/relationship to career, and teachers' knowledge and beliefs related to computing. In addition, open-ended questions were included to further understand their conceptualization of CT, the role of technology in teaching computing, teaching strategies for supporting computing integration, and an opportunity to share any final thoughts.

Semi-structured interviews (Merriam & Tisdell, 2016) with each participant were conducted twice during the study with each participant. An initial interview took place following completion of the survey, and a second interview occurred following completion of observations of their teaching/classrooms (see Appendix G for an example

of questions). These interviews were done virtually via Zoom, a video conferencing platform, and were recorded. First interviews were used to follow up on survey responses, dive deeper into their computing content knowledge, learn more about the technologies they use (if any) to support computing integration, and pedagogical choices. Second interviews were used to follow-up on observations, ask about their motivation for teaching CS/CT, learn more about CT/CS assessments (if any), learn more about the role of equity in their practice (if any), their beliefs about students' abilities in computing, and ask any lingering questions. Interview questions were aligned with the TPACK-CT framework (i.e., content knowledge questions, pedagogy questions, technology questions) and aided in deepening my understanding of the teachers' perceptions and approaches of integrating computing into their content areas.

At least three hours' worth of observations of each teacher in their classroom occurred. These were coordinated with the teacher when they felt they were teaching a lesson that integrated CT/CS into the content, and could occur during consecutive classes/lessons or multiple visits. A protocol aligned with the TPACK-CT framework and guided by other TPACK observational protocols was developed to guide fieldnotes during the observations (Bender et al., 2015; Lee et al., 2006; Piburn & Sawada, 2000) (see Appendix D) to aid in identifying the different constructs within TPACK-CT that were visible.

Teachers were also asked to complete reflections following their lesson(s) based on the questions used by Mouza and colleagues (2017) in their work with pre-service teachers. These guided reflections (see Appendix H) could either be audio-recorded by

the teacher or written down as they reflected on their approach to the lesson (how they introduced it, what occurred during the lesson, what strategies they intentionally used, how explicit was computing, any supporting materials used in lesson creation, etc.) and a reflection on how it went (did the lesson support students' development of computing skills, take-aways, things they would do differently, etc.). During the first interview, we discussed the reflection protocol and the teacher's preference for how to complete them as options were provided like handwritten in a notebook and scanned into Google Drive via a photo, voice memo on phone or computer, typed in a Google Doc/Word Doc, or recorded onto a VoiceThread that I share with them. After each observation, I also reminded them to take 5-10 minutes to reflect on their lesson and re-iterated instructions to complete this reflection - all teachers chose to complete these by writing (either in a Google doc or an email). These reflections were integral in discovering teachers' own perceptions of their TPACK-CT and acted as a tool to see how my own interpretation compared.

A secondary data source, curriculum artifacts, was also collected. These included hand-outs used during the lesson, instructional materials, teacher notes, images, etc. These artifacts supported further interpretation of teachers' TPACK-CT and facilitated more detailed sharing of specific lessons that were used. Table 5 shows the total number of pages, hours, and images collected throughout the study.

Table 5

Totals of Pages, Time, Images, and Artifacts Accumulated During Data Collection

Total # of pages	Total hours & minutes	Total images & artifacts

(Gathered from surveys, interviews, observations, reflections, artifacts, and analysis notes/memos)	*(Accumulated during interviews and observations)*	*(Gathered during observations and from artifacts)*
251 pages	16 hours and 12 minutes	521

Data Analysis

This comparative multicase study required two stages of analysis; the first stage was the within-case analysis (bounded by the teacher) and the second stage was the cross-case analysis (across the teachers) (Merriam & Tisdell, 2016). First, each teacher was treated as their own case and their individual data was analyzed to answer the research questions. After creating individual case narratives/portraits for each teacher, I used these narratives to begin the cross-case analysis of the teachers (see Table 6).

Table 6

Overview of Data Analysis Stages

1st Stage Within-Case Analysis		**2nd Stage Cross-Case Analysis** *Informed by the 1st stage*	
Data Used:	Analysis Processes:	Data Used:	Analysis Processes:
❑ Survey ❑ Interviews ❑ Observations & fieldnotes ❑ Teacher reflections ❑ Artifacts	❑ Descriptive statistics for survey ❑ Thematic analysis of open-ended survey questions ❑ Qualitative content analysis	❑ Case portraits from 1st stage ❑ Codes, categories, & themes developed in 1st stage ❑ Survey results	❑ Data coded to align with the research questions ❑ Grouping open codes → analytical codes = thematic findings

	❑ Constant comparative method		
● **1st** - Each subunit (teacher) treated as own case = 3 cases ○ Individual portraits: ■ Context + initial survey findings to document confidence in CT ■ Computing experiences ■ Lessons taught + reflections		● **2nd** - Align codes to research questions ○ Uses individual portraits ○ Results in thematic findings across cases ○ Compares survey results ● **3rd** - Answer research questions	

After cleaning the survey data (replacing names with pseudonyms, aligning the Likert scale values so that the scale went from 4 = strongly agree and 1 = strongly disagree), descriptive statistics was used to analyze the Likert-items since Likert scale ratings were used to describe teachers' perception of their own relationship to, beliefs regarding, and confidence in teaching computing. The recruitment survey and the more in-depth survey taken by each teacher was used to inform case portraits for each participant, and provided both context and detail about each teacher's conceptualization of CT and their beliefs regarding computing and integration. Their survey responses were also open-coded to align with TPACK-CT categories to visualize their conceptualization of CT (content knowledge) - including technological and pedagogical strategies, and were used to show any similarities and/or differences across participants during the cross-case analysis. Analysis of all qualitative data was inductive, including the open-ended survey responses, used the constant comparative method (Hatch, 2002; Merriam & Tisdell, 2016; Miles & Huberman, 1994), and was grounded in and framed by the

TPACK-CT framework (Angeli et al., 2016; Koehler et al., 2013; Mouza et al., 2017) and the ISTE and CSTA's conceptualization of computational thinking (CSTA & ISTE, 2011).

I focused on one participant at a time as open-ended survey questions, fieldnotes, interviews, and reflections were reviewed and organized/coded to align with each research question and the TPACK-CT framework. For instance, if a teacher used Bee-Bots in a lesson on sequencing a story, the open code "bee-bots" might fall under the category "TK-CT" and the code "sequencing a story" would be categorized under "CK-CT." Each of these headings/categories also aligned with a research question. The TPACK-CT helped construct category headings to group and align each code (e.g., CK-CT, TK-CT, PK-CT) as well as kept the codes aligned with the research questions. This process was not linear and happened multiple times throughout the study (e.g., right before the first interview, again before the final interview) - each set of data and its corresponding codes built over time, as more data was added.

After all data collection was completed and grounded in the research questions, the collected answers, notes, transcripts, and reflections were analyzed again using qualitative content analysis (Saldaña, 2013) - with the focus first on each individual participant (e.g., I only focused on Mr. Pi until his case portrait was completed before I moved to another participant). Open coding processes were followed by grouping open codes into categories, and resulted in thematic finding - all of which were organized according to TPACK-CT constructs within coding tables (Harry et al., 2005). These categories and codes resulted in multiple thematic findings that were detailed in each case

portrait to help understand what each participant's TPACK-CT looked like when enacted in the classroom - especially focusing on how they conceptualized CT knowledge in the classroom (CK-CT), what pedagogies they enacted when teaching CT/CS (PK-CT), and what technologies they used for teaching CT/CS (TK-CT). Since teaching is so nuanced and complex, many of these findings overlapped and interacted with each other resulting in a more vivid picture of how each participant worked to integrate CT/CS into their classroom, their enacted curriculum. The information gathered from the surveys was also weaved in during interpretive coding and case construction processes.

Through their reflections, interviews, and fieldnotes from observations, each individual case narrative shares the context of the teacher (e.g., grade level, years teaching, experience with CT, info given during interviews, etc.) woven in with initial survey findings on confidence/knowledge of CT and highlights of lesson(s) they used to integrate CT/CS. Primary data was prioritized; while secondary data was used for triangulation purposes in order to increase the accuracy and credibility of findings. All names are pseudonyms.

The cross-case analysis looked across each of the case portraits and coding tables of each participant to identify themes and key differences/similarities. Coding tables were also re-evaluated and codes were grouped and replaced with conceptual categories to better encapsulate shared findings (e.g., "BeeBots," "Ozobots," and "Raspberry Pis," were collapsed to "robots") so that word clouds and word counts could be created as another way to compare, inform, and validate my cross-case thematic findings. The codes in each category were input into word cloud software and counts were taken to see if it

supported the resulting cross-case themes and to provide more insight. The cross-case analysis process was inductive with the goal of building "abstractions across cases" (Merriam & Tisdell, 2016, p. 234).

Trustworthiness

Various strategies were used in this dissertation to ensure the trustworthiness of the data, or its credibility/internal validity, dependability/reliability, transferability/external validity, and confirmability (Lincoln & Guba, 1985; Merriam & Tisdell, 2016). While there is still debate over how to best discuss trustworthiness when it comes to qualitative studies versus quantitative research, I will focus on the terms more often used in conjunction with qualitative research as they better align with qualitative assumptions: credibility, dependability, transferability, and confirmability. Credibility, or the extent to which my findings are trustworthy, strategies included triangulation and member checks (Merriam & Tisdell, 2016). Triangulation involved collecting multiple data sources in order to compare them; for example, looking at the teacher reflection compared to the fieldnotes and seeing if there are discrepancies or differing interpretations. Member checking was also used for ensuring credibility as drafts of each case portrait were shared with the corresponding teacher. Teachers were asked if they felt their case portrait was reflective of their experiences and/or if there were concerns or inconsistencies. Mrs. Bee said she found it to be a detailed and thorough summary with no concerns or corrections. Mr. Ozo also said he saw himself in his case portrait and made me aware of one inaccuracy regarding the school's broadband access (he clarified

that many students and some teachers did not have broadband at home, but that the school did) and this was fixed in the case portrait. Mr. Pi did not provide any feedback.

Dependability, or the extent to which the findings are consistent make sense, (Merriam & Tisdell, 2016) was enhanced through an audit trail when I shared how the study was conducted and how the findings resulted. Sharing my research positionality and strategies to mediate it, as well as through the peer examination that is inherent in the dissertation process, are additional strategies to enhance my study's dependability. To support transferability, I ensured my dissertation provided thick and rich descriptions. Finally, documenting my research processes to aid others in tracing the findings back to the original data sources supports confirmability, or an effort to mediate my biases and positionality.

Summary

Using a comparative multicase approach, alongside the TPACK-CT framework, I aimed to explore and understand the experiences of Minnesota teachers working to integrate CS/CT into their classrooms. After recruitment, three participants filled out a survey related to their conceptualization of and confidence in CT content knowledge, participated in two interviews, were observed, completed teacher reflections on their lessons, and allowed me to collect curriculum artifacts. Using an inductive, constant comparative method and qualitative content analysis, I created rich case portraits of each participant's enacted curriculum to answer the first research question and its sub-questions. To answer the second research question, a cross-case analysis was done.

The next four chapters share the findings from my analysis. Chapters 4, 5, and 6 are the resulting case portraits of Mrs. Bee, Mr. Ozo, and Mr. Pi. These chapters share the enacted curriculum of each of these teachers organized by the TPACK-CT framework to better answer the first research question and its sub-questions. Chapter 7 presents the cross-case analysis that answers the second research question as it shares the thematic findings that are visible across the three cases. Note that none of these findings and chapters aim to suggest certain curricula, PD experiences, or strategies are better or worse than others, but instead illuminate the perceptions, experiences, and realities of Minnesota teachers who are working to bring computing into their classroom. So while their views and beliefs at the time of the study are shared, it should be acknowledged that they can also change over time or the information may not be 100% accurate as they, too, are human (e.g., “CS AP” may be said but they could have actually meant the“AP CS A” course specifically).

CHAPTER 4: MRS. BEE

"*I'm not the computer teacher.*" - Mrs. Bee

Figure 7. Mrs. Bee's introductory ID card.

Introducing Mrs. Bee

Mrs. Bee was a STEM specialist for kindergartners through fifth graders during the time of this study (2021-2022) and had been teaching for over 27 years. Her background is in elementary education and early childhood with an integrationist certification; she has experience as an interventionist for math and reading, and as a classroom teacher for first, second, and third grades. Most of Mrs. Bee's career has been spent in the same district in a variety of roles, with her current STEM specialist role beginning a little over eight years ago. The supportiveness of her colleagues, administration, and community are evident in the way she talks about how the kindergarten teachers supported her when she was a temporary classroom teacher during/because of COVID-19 (2020-2021), reflects on her principal's support in

transitioning to the new role of STEM specialist eight years ago, and discusses the administration and community's investment in supporting her classroom and donating supplies.

Her district has many resources and is composed of an elementary school, a middle school, and a high school. The district is located in a community of less than 10,000 people and serves more than 1,500 students from two suburban areas right on the outskirts of a major Midwestern city. The district's only elementary school serves 700 students with the following demographics: 62.9% white, 13.1% Black, 8.1% Asian, 8.6% two or more races, 6% Latino/a, and 1% American Indian or Alaska Native (U.S. Department of Education). Mrs. Bee works in the district's only elementary school where, as a STEM specialist, she sees every single kindergarten through fifth grade student once a week for 15-30 minutes.

The STEM program and its Lab were originally established by a colleague of Mrs. Bee around fifteen years ago when the administration wanted to brand the school as a "STEM school," and invested in building up the program. Mrs. Bee was a teacher in the district at the time and had the impression that the superintendent was very motivated to adopt the "next big thing" to set their school apart. Mrs. Bee transitioned into the STEM specialist role when the original founder left the school. While the school is no longer known as a STEM school because classroom teachers lack the time to integrate it into their classrooms, the STEM program and specialist position have remained.

Classroom and Computing Context

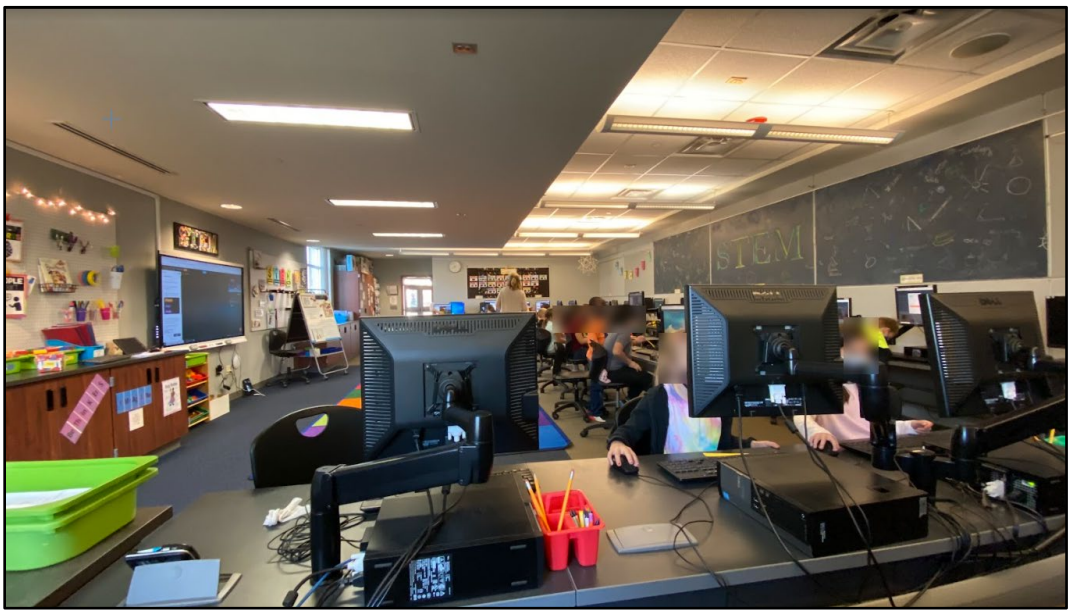


Figure 8. Mrs. Bee's classroom arrangement.

Mrs. Bee's classroom, or the STEM Lab, houses 32 desktop computers, cabinets and shelves, books, a 3D printer, and lots of supplies like dominos, toilet paper tubes, etc. She began integrating computational thinking concepts into her classes around 2015 and tries to incorporate computing and/or CT activities at least two to three times per week using lessons that are 30 minutes or less. She has pursued her own professional learning experiences to grow her CT understanding since her school has not brought in any CS or CT-focused professional learning experiences, and since she does not have a lot of overlap (both in their content and time together) with those in similar technology instructor roles at the middle and high school. She learns through being both a participant and a volunteer. She has spent a lot of time doing informal learning via Facebook groups, searching out resources, volunteering at community coder dojos, getting additional certifications through higher education institutions, and participating in the one

professional development experience she found within her state related to computer science: a state-wide professional development cohort focused on supporting educators in providing equitable and engaging computer science learning opportunities. These learning experiences combined with her role as the science lead for the school have informed how she approaches integrating CT/CS into her curriculum.

Despite Mrs. Bee's passion for teaching, integrating CT, and work to pursue learning opportunities, there are multiple challenges that come with her role. One of the challenges Mrs. Bee faces as the STEM specialist is that her state does not provide STEM standards, so she instead uses the science standards to make sure her computing activities are also aligned with those standards. While in the past there have been initiatives within her school to involve classroom teachers in STEM (e.g., all K-2nd grade teachers were given Legos to use along with STEM activities in which to use them), they often fade away as more things are added to classroom teachers' plates and the school pursues other initiatives. In addition, Mrs. Bee must spend a lot of time laying the foundations of using technology for her students - how to log in, how to use Google Classroom, etc. - and battling the school's technology-use policies' interference with what they can do in the classroom (i.e., applications are blocked, projects cannot be shared because emails are locked down, etc.). During COVID-19 in the 2020-2021 school year, no student received STEM because she was needed to serve as a full-time kindergarten teacher. Missing a year of STEM meant that during the 2021-2022 school year, some students had never been in her classroom before, others had forgotten the classroom culture and norms, and everyone had to adjust to being in a classroom setting together. Finally, she noted that it

is hard to know if integrating computing into her STEM classes influence students' desire to pursue CS. She knows that while some students can get enrichment at home and engage in computational thinking concepts and computer science tools beyond her classroom, not all students have those opportunities and it is hard to track if students continue to pursue computer science beyond elementary school:

> My district, probably like a lot of [districts in this state], we don't really have like a computer programming class. There's a STEM class at the middle school that's an elective and I know they do a variety of things similar to what I do…but I feel like it [computer science opportunities] dwindles after eighth grade…Maybe my sixth graders are doing awesome things independently, but yeah, I mean who knows if they'll give it up [as they leave the school]. (second interview)

The Enacted Curriculum of Mrs. Bee

In order to better understand the enacted curriculum of integrating computing in Mrs. Bee's classroom, TPACK-CT categories are used as a guiding and organizational lens. How she conceptualizes CT knowledge in the classroom (CK-CT), what pedagogies she enacts when teaching CT (PK-CT), and what technologies she uses for teaching CT (TK-CT) are further expanded upon. Since teaching is so nuanced and complex, many of these categories and findings overlap and interact with each other resulting in a more vivid picture of how Mrs. Bee works to integrate CT into her classroom.

Mrs. Bee's TPACK-CT

Mrs. Bee's years of experience, personal experiences learning computational thinking, role within her school, emphasis on relationships with students, and supportive

relationships with(in) her district all provide some of the context that uniquely influences her TPACK-CT within the classroom and in her role. The feeling of being free to try different things allows her to seek out new learning, build on her resources and own foundational learning, and modify existing curriculum she encounters. Her willingness to learn and try new things is evident in one of her interviews when she shared that she really tries "to do different things to introduce…computational thinking….I'm just kind of constantly trying to build this in…and look for unplugged ways to teach computational thinking." Since Mrs. Bee does not conceptualize CT as requiring computers, she does not rely on them to integrate CT. One of her pedagogical strategies is building on lessons and prior knowledge, so the more CT concepts and practices she can introduce through unplugged lessons, the more aligned it is with her project-based learning approach. This strategy also supports her students in building a foundation in CT prior to introducing the more advanced technology (i.e., robots, computers, etc.) in order to avoid potentially being overwhelmed with the technological possibilities. Computational thinking is something she intentionally builds into and aligns with her units and lesson planning as she believes it expands the possibilities for student learning. Despite the challenges (i.e., time, behavioral challenges post-COVID19, logistics, technology barriers, etc.) and dwindling computing pathway available to students beyond her classroom, she hopes students are getting long-term exposure as she believes every student leaves with basic programming skills and a belief that they can be computer scientists. Her beliefs and experiences are evident in her pedagogy, technology integration, and content knowledge – as well as how all of these weave together to create her TPACK-CT teaching tapestry.

Conceptualization of Computational Thinking [CK-CT]

Mrs. Bee shared on the CT Survey at the onset of this study that she was confident in her understanding of computational thinking and views CT as "a series of steps and problem-solving methods similar to the way a computer would think or operate. Some of these steps include breaking down a problem into smaller chunks, looking for patterns and using algorithms" (survey response). While her first encounter with computing was when she had to quickly learn Scratch for her new role, her view of CT continued to be shaped by the professional learning opportunities she participated in, and she feels computational thinking is now woven into most of what she does. In addition to teaching students how CT is a problem-solving process, she creates experiences for them that highlight the interdisciplinary nature of CT's application to other subjects, particularly science. She focuses a lot on developing CT skills like decomposition, pattern recognition, algorithm identification and writing, and helping students understand what it means to "think like a computer."

During one of our interviews, Mrs. Bee talked about the way that explicit connections are helpful for students and the ways in which coding makes CT more visible. During her ongoing lessons/projects, she often tries to help students make the connection between what they are doing and CT by using vocabulary. For example, she'll relate how a story has a beginning, middle, and end just like a sequence for a computer does. In addition, she shared that she believes many elementary education teachers are also incorporating elements of CT, whether knowingly or not, by pointing to the multi-step directions that are often used in classrooms or the making and building that often

happens in elementary classrooms -- the students and teachers may just be missing the explicit connections. In order for teachers to be able to point out connections between CT and what they are already doing in the classroom, they need training and support to understand and learn about CT. For instance, she shared how a recent student teacher knew nothing about computational thinking. Yet Mrs. Bee notes how teacher preparation is a perfect place to make CT more visible because of the overlap of CT with other subjects; she believes that CT can be studied alongside state standards. She pointed to the overlap with math curriculum (e.g., patterning) and how well computing fits into science and engineering content. As the science lead for her school, she often starts with the science standards as the backbone of her lessons and projects, and then integrates and highlights computing concepts into the final project she has students work toward within the unit. This both helps ensure her lessons are aligned with current curricula and provides opportunities for students to practice and grow their computational thinking.

Despite the inherent CT concepts already in elementary education content areas and pedagogies, when asked about her beliefs around her students and CT, Mrs. Bee shared she believes more second through fifth grade students (80-90%) can program than work through CT (75%). She pointed out that she believes there is just a disconnect and/or disinterest for some students. However, she still works toward the goal that by fifth grade all of her students will be able to write a simple program that includes basic instructions to result in an action(s) to occur in sequential form.

Pedagogies for Teaching CT [PK-CT]

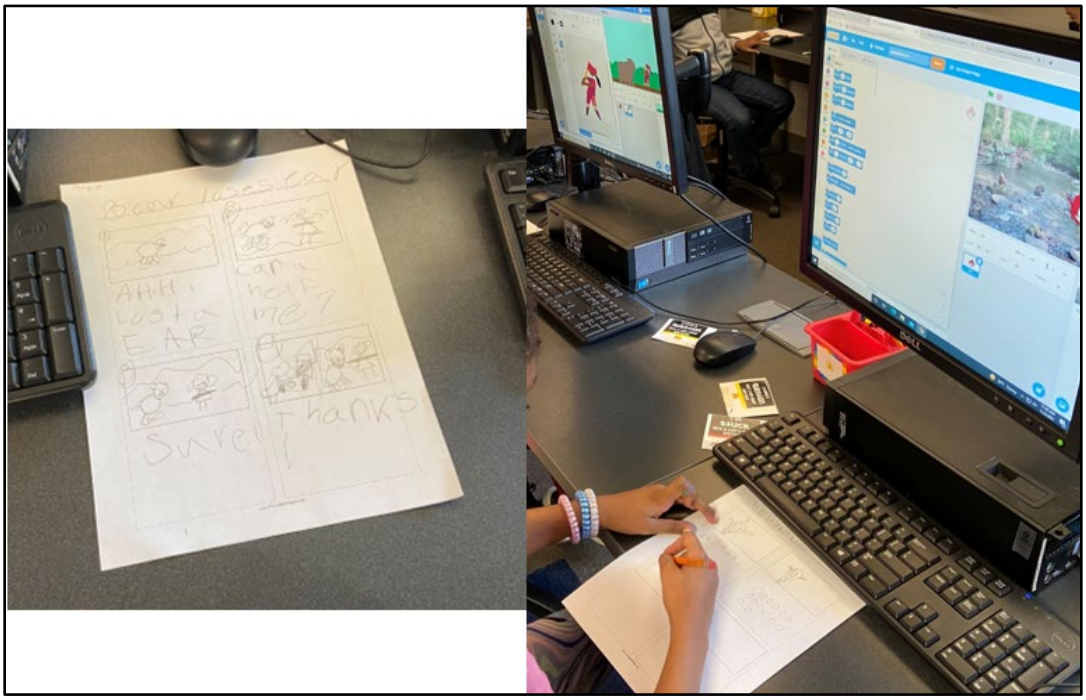


Figure 9. Student work examples. A student works on their storyboard (right) and a student's completed storyboard from the prosthetics unit (left).

The pedagogical choices Mrs. Bee makes teaching CT include ensuring activities are hands-on, incorporating reflection, building on activities/technology which lead to a culminating project, and emphasizing the importance of planning/storyboarding while still leaving room for students to tinker and pursue their interests. She shares that she personally feels successful when her students get to do hands-on activities (post-observation reflection). She also focuses on helping students make connections (with vocabulary, prior content, CT terms, other disciplines, etc.) and providing varying levels of depth that students can choose to pursue. For instance, whenever she incorporates books and stories, she uses them to not only spark students' interest and introduce a topic/content area, but she also uses it to showcase the connections between a story being

composed of multiple steps with the concept of a computer "thinking in order." Overall, she views relationships as most important and this is evident in the individualized and project-based choices she provides students.

Just as Mrs. Bee believes that many CT principles are inherent in elementary school, she also consciously tries to build them into her lessons. She explained, "I try to use a variety of hands-on design challenges that cause students to use problem-solving skills, as well as teamwork, to determine the best way to solve a problem. The challenge will lead to a product they share." Her goal is to integrate CT into everything she does. This approach is visible in the way that she designs her units and how she makes connections to other subjects, such as science. For example, with third and fourth graders she used story books to introduce them to prosthetics. Instead of stopping with a book and reviewing vocabulary, Mrs. Bee then had the students work toward creating their own Scratch story about an animal needing a prosthetic limb - not only did this help them get comfortable with using Scratch, but it also allowed them to try their hand at programming while also being able to create their own story. In her reflection on integrating CT into this lesson she shares, "I think using Scratch is very engaging for the students. Learning how to create a digital story with the coding blocks helps them to see the sequence of the computations. It also ties to language art skills." She also taught a unit on natural disasters and began with having students use Legos to build solutions to solve a natural disaster problem (e.g., a bridge after an earthquake). Eventually, the students received a challenge card where they worked together to build something that their Edison robot could use to overcome the challenge. Not only does this align with the

science standards, but each lesson builds on the next and there is a mixture of unplugged activities and customization. Her approaches to introducing and teaching computational thinking varies depending on students' abilities and she notes that their interest also impacts their level of involvement and participation.

Figure 10. Materials to introduce a lesson. The left image shows one of the books used to introduce prosthetics, storyboard handouts, and printed out Scratch directions for students to use as they get to work on their project. Mrs. Bee models how to fill in the storyboard (right image).

Including unplugged ways to teach computing is another deliberate pedagogical choice - Mrs. Bee knows that a lot of students see her classroom as a device-centric room since there are many computers, but she also finds value in letting kids tinker and be hands-on in the physical space. Mrs. Bee uses a lot of multi-step directions in her lessons since the activities often build on each other, and she articulates to students that this connects to how computers think - she explicitly helps make these CT connections. There is also a lot of repetition, redirection, reinforcement, and visuals employed throughout a

lesson. Since Mrs. Bee works with all elementary grade levels at her school, students have varying degrees of comfortability with the technology in her room. In response, she ensures she's building on concepts, includes a planning stage, and tries to tie back lessons to what the students have been doing. She also utilizes classroom culture tools like many different laminated print outs with key steps, signs about logging in, and even a process for asking questions (cards they can stick to their monitors saying they are "stuck and need help," "stuck but can work on something else," and encouragement to ask their neighbor). In addition, after introducing the day's activity, she spends her time walking around the classroom to answer students' questions, reminding them what to do, and being shown the work students have completed thus far.

Figure 11. Mrs. Bee helps students as they gather around to see a students' project.

Mrs. Bee wants her students to see themselves as programmers so she emphasizes the computing skills that they are using throughout lessons. In addition, she likes to talk

about and share different STEM careers to students. She believes that CT ultimately makes things more fun and helps bring the "technology" element into Science Technology Engineering and Math (STEM) curricula. The strategies she employs to teach CT help students go deeper while connecting to prior learning and allowing students to have fun. To help students go deeper, she tries to show a lot of examples and models computational thinking for them. She shares in a post-observation reflection, "One strategy I planned to [and did] use was talking through step-by-step directions with the BeeBot. By modeling with the BeeBot different ways I would use it, students could predict and see how my own thinking worked out." Despite her passion for CT and the explicit connections she calls out, Mrs. Bee struggles with how to get students to verbalize their own computational thinking and does not currently have ways to assess CT. However, she does try to incorporate students sharing their final product, but does not feel there is always a lot of talking about/through it.

Technologies for CT (TK-CT)

Similar to her pedagogical strategies to teaching CT, her approach to incorporating technology into teaching CT is intentional, student-centered, and enriching. Her focus on the students is visible in the ways she incorporates age-appropriate visuals, looks for and celebrates their enjoyment, ensures there is a variety of activities, and designs project-based curriculum. Her pedagogical choices for weaving in technology often include encouraging students to participate in "structured play" via different stations, designing activities for collaboration, promoting students' own customization, and centering students' voices and choices. While she identifies that technology is both a

barrier and a resource for integrating CT into the classroom, she uses a variety of strategies to help students get the most out of it (i.e., supporting directional materials, repetition, modeling, peer help, etc.). For example, she shares in a post-observation reflection:

> Every time I use Scratch I find myself trying to reinforce that everything happens in an order after they click the green flag. I think it's hard for them to understand the top-to-bottom nature of the sequence since they are used to a left-to-right progression with reading.

Ultimately, she sees bringing in technology as a strategy to make CT choices more visible for students while they are also having fun.

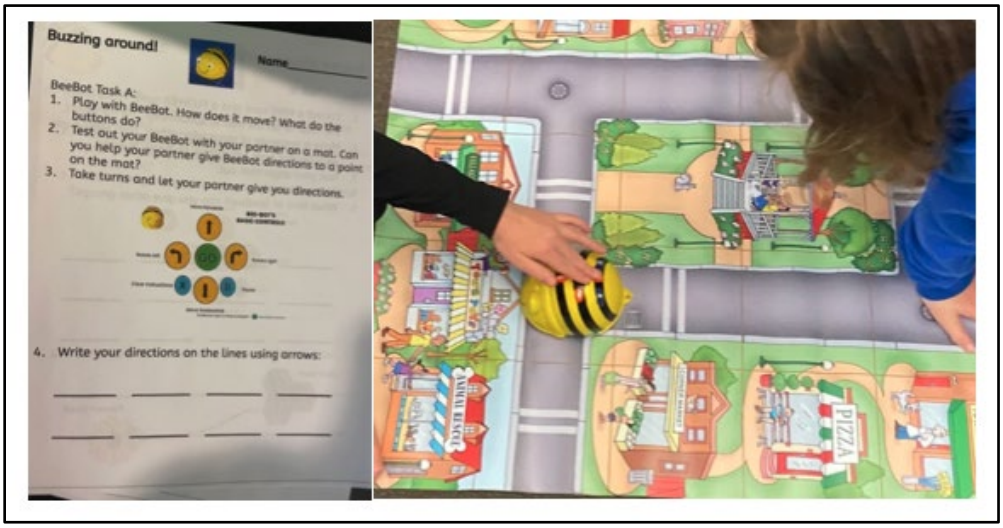


Figure 12. Mrs. Bee introduces Beebots to the class. Students are given a guiding worksheet to play with the robots in their small groups.

As the STEM specialist for her school, Mrs. Bee has access to a lot of technology including computers, robots, 3D printers, iPads, and more. However, she purposefully chooses and teaches the basics for applications that students will encounter throughout their time at the school, as well as using technology that enhances the connections

between subject matter standards and computational thinking practices. A challenge to using technology to build on computational thinking skills is that students first need fundamental computer skills - including digital safety; this includes basics like logging in, accessing their school email, typing, logging out when they are done, and more.

As students build on these skills, then she begins weaving in tools that integrate CT more; some of these technologies to specifically integrate CT are Scratch (Jr.), robots (Edison, BeeBots), Google apps, and physical building materials like Legos. Just as she makes connections in content and CT, she also does that with the tools she has students use (e.g., Scratch is not just coding but also for art). She uses technology to help students: identify and write their own sequences (i.e., a Scratch story, stop motion, programming a BeeBot, etc.), make connections to other subjects (e.g., art, science, and math), create their own final projects that weave in prior knowledge, deeper understanding, and CT connections, and to have fun. While she asserts that most of what she does with technology and CT is simple and block-based programming, she appreciates the exposure and enrichment using technology combined with CT provides her students.

While Mrs. Bee does not think computers are necessary for learning computational thinking, she does believe it can grab students' attention and help them make connections between CT and what they are learning. To help alleviate students being distracted by the many capabilities that technology can provide, she provides a lot of (re)direction and always emphasizes the planning phase prior to accessing the technology. Despite the potential distractions, she sees how incorporating technology into her computing lessons and projects engages students. She also recognizes how the

availability of technology continues to grow each year and sees it as a way to get students "hooked" or interested.

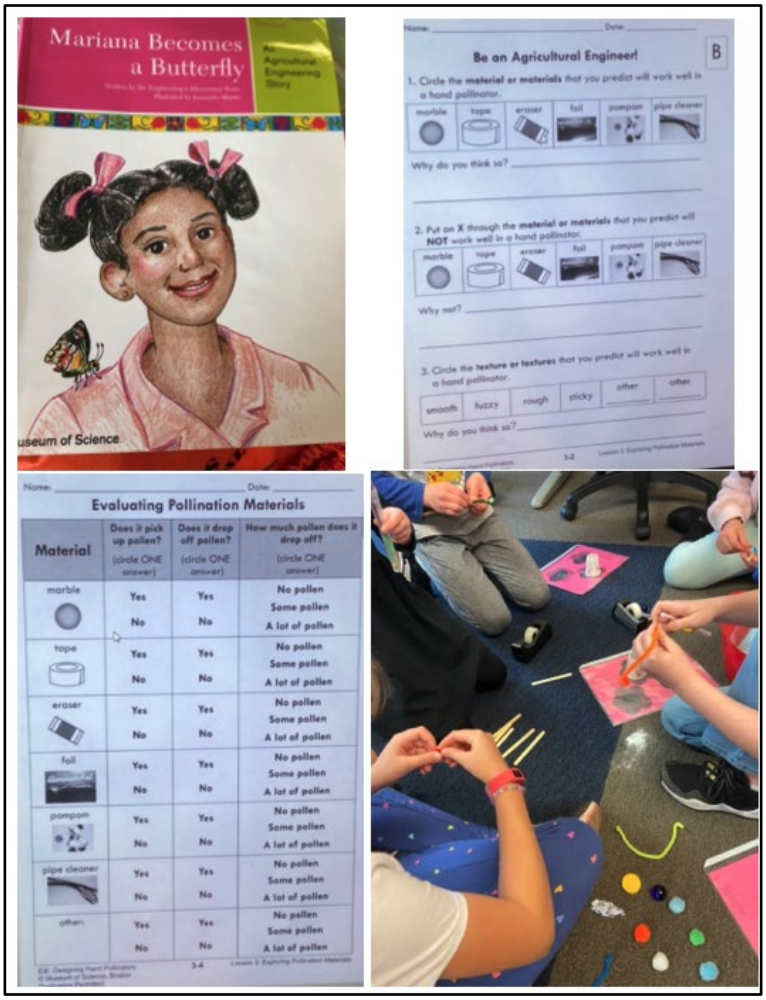


Figure 13. Materials and lessons to introduce agricultural engineering. Students learn about agricultural engineering via a story, and then use a worksheet about pollination to fill out their hypotheses around pollinating materials. Next, they create hand pollinators to test their guesses. During the next lesson, they do a pollinating themed activity with BeeBots.

Technology's definition incorporates a broad variety of mediums, so when reflecting on her own technology usage, Mrs. Bee makes sure to mention more hands-on tools like Lincoln Logs, Legos, 2x4s, Duplo bricks, and other building materials. She finds that bringing in tools helps make up the "T" in STEM and that it connects well to

science, engineering, and math. She sees technology as a way to also aid in decomposing and demonstrating that CT concept. In a post-observation reflection, she shared that she likes BeeBots because "it is a simple way to get students to think algorithmically. To complete the task, students had to break down the problem, look for patterns, and develop a step-by-step solution for their partner."

A challenge that Mrs. Bee faces is having the iPads managed by the district - this often means the apps are pre-selected and are not the ones she wants to use with students. While the apps may be the ones classroom teachers prefer, they are often not the ones she would have chosen. This and other school restrictions (e.g., students cannot email projects to family or themselves because their emails are locked down) can be additional obstacles that she must overcome in order to thoughtfully integrate technology with CT.

Summary

As an elementary school STEM specialist, Mrs. Bee does not have state standards that guide her curriculum, although the science standards do help guide her planning. This also means that she is designing an array of lessons since she sees every grade level in her role and she must also include time to teach foundational computer skills. Despite these challenges and others, her learning experiences, as well as feeling supported by her administration, inspired her to integrate computational thinking into her classroom. Knowing that her role allows her to be flexible has enabled her to try new things and continue to build on her curriculum by incorporating unplugged activities to specifically help her students build CT skills.

Mrs. Bee emphasizes the computing skills students are using as they program and introduces them to many STEM careers in hopes that they see the possibilities that are open to them. Since Mrs. Bee sees the interdisciplinary nature of CT, she feels comfortable helping her students make some of those explicit connections as she teaches across and weaves together multiple subjects. One of the ways Mrs. Bee approaches CT integration is through hands-on activities and lessons that build on one another to eventually culminate in a project that brings all of the prior learning and foundational concepts together. In addition to the practical strategies Mrs. Bee uses in her classroom like visuals, modeling activities, and concise instructions, students are also often given multiple options to align with their interest, time to tinker and explore their ideas, and encouraged to collaborate.

While Mrs. Bee acknowledges the barriers technology can present, she also sees it as a resource to make visible students' CT in action. Mrs. Bee uses technology in her classroom when integrating CT through a mixture of unplugged activities, stations for students to play, and incorporating it into student-centered projects. Technology is also a tool that Mrs. Bee uses to help her students make connections across subjects, CT, and their learning, as well as a key way to engage students and get them interested.

Mrs. Bee believes that by the time all of her students leave the elementary school they will have experienced CT/CS and that most would even consider themselves computer scientists, but she is unsure what computing opportunities will await them. While she has two teachers she would consider her "job alikes" at the middle and high school, their curriculum looks much different than her own since their courses tend to be

more industrial arts and workshop-focused (e.g., building a car). In a state without CS/CT standards, students' computing pathways and exposure are dependent on individual districts, schools, and teachers deciding to integrate it. So when asked what advice Mrs. Bee would share with a teacher interested in beginning to integrate computational thinking she shared,

> ...don't reinvent the wheel, you know, don't try to create it on your own… Go searching for other lessons that are already out there…start with that and see what other people are already doing. And then also look into how does it align with your current units…[like] into a science or an engineering unit that you're already teaching. (second interview)

CHAPTER 5: MR. OZO

"*My kids get a little bit of coding. My neighbors' [kids] don't.*" - Mr. Ozo

Figure 14. Mr. Ozo's introductory ID card.

Introducing Mr. Ozo

Mr. Ozo was a fourth grade teacher during the time of this study (2021-2022). He has taught fourth grade for fourteen of the seventeen years he has been teaching. Before becoming an elementary school teacher, Mr. Ozo studied and worked in physics, material engineering, and automotive electrical engineering. Both he and his wife decided to leave the engineering field to become elementary school teachers through a graduate program. At the time of this study, they both taught for the same district in his hometown.

Since at least forty percent of students qualify for free or reduced-price lunch, his district is considered Title 1, which means the district receives federal funds to support academically at-risk students through educational services. The district is composed of four elementary schools, one middle school, and one high school, which are all located in

the largest town in the county (population of less than 19,000). Mr. Ozo finds the town to still have a small community feel. The boundaries for the school are known to change, but it has an open-enrollment dynamic allowing students to take the bus into the town from more rural areas and/or to have access to resources (e.g., special education services (SPED) are provided for at the school in town while it may not be offered in a more remote area). His elementary school had around 400 students enrolled during the 2021-2022 school year with the following demographics: 62.9% white, 21.3% Latino/a, 5.9% two or more races, 4.9% Asian, 4.2% Black, and less than 1% American Indian or Alaska Native (U.S. Department of Education).

Mr. Ozo had 25 students in his fourth grade class with about one third of them receiving SPED services during the time of the study. There are two other fourth grade teachers in his building that he works closely with, as well as small professional learning communities (PLCs) in the district that allow for additional collaboration. The teachers in his school wear a lot of hats and roles since the school does not receive large funding awards. For instance, teachers had to open their rooms thirty minutes prior to school starting during the 2021-2022 school year because of a change in bus schedules as result of COVID-19 - they could not take attendance during this time, but needed to have something for the students. Mr. Ozo elected to make this a time for students to engage in computer science.

A few years prior to this study, Mr. Ozo originally began to do computer science (CS) with students during recess, before class, and sometimes during class as a result of participating in a year-long, state-wide CS/CT-focused professional development cohort

focused on supporting educators in providing equitable and engaging computer science learning opportunities. The year after his PD cohort ended, he launched a coding club that eventually grew to have over 90 students and involved three other teachers that provided support for the club. The coding club lasted three years until COVID-19 caused the school to go fully online. Once in-person classes resumed, Mr. Ozo began incorporating computer science during the thirty minutes prior to class while students ate their school-provided breakfast (see Figure 15). Mr. Ozo intentionally tries to integrate CT concepts every day for fifteen minutes or less, but by the end of the study felt that he actually integrated it more than he first acknowledged.

Figure 15. Students work on their Scratch projects in Mr. Ozo's classroom before school begins.

Classroom and Computing Context

Students sat on a variety of chair styles as they gathered around different sized tables located along the edges of Mr. Ozo's fourth grade classroom to begin their day (see Figure 16) during my visit. The middle of the classroom was also a learning and

gathering space since the tables were organized around the perimeter of the room. At the front of the room were whiteboards and a Smart TV where music and videos often played, and examples or models were shown. Despite that not all students and teachers have broadband access at home, all elementary students at Mr. Ozo's school have Chrome computers - these are open at their desks during breakfast while they work on their Scratch or CS projects, alongside a morning routine checklist.

Figure 16. Mr. Ozo's classroom arrangement.

While someone coming into Mr. Ozo's classroom today would experience an elementary school day covering a lot of content with implicit and explicit CT connections seamlessly woven in, Mr. Ozo explained it had not always been that way. For Mr. Ozo, connecting classroom curricula to science and engineering had felt natural because of his background, but he "didn't see [himself] as a computer scientist" so that was not a connection he felt he could make. His introduction to computing was about four to five years before the study when it had "to be explicitly thumped into [his] head" while

participating in a year-long, statewide, inclusive computer science education learning cohort that his media center colleague introduced him to. This PD experience shifted and expanded his previous beliefs about what coding is and how early students can learn to code.

Participating in this learning cohort both introduced him to a variety of people from throughout the state and also exposed him to block-based coding tools like Scratch, as well as the language to articulate and recognize computational thinking concepts. He and his colleague in the cohort became technology coaches (on top of their full-time teaching load) and over time were able to get the middle school tech coach and media specialists to participate in the same cohort. This PD experience, along with the support of his school leadership, and knowing that others were working on bringing in CT were foundational to his integration of CT/CS.

Mr. Ozo's original coding club had both students and the community engaged. The club grew in size over time to the point where the coding club expanded to three schools in the town. He found that most of the community either wanted CS (which they often equated to coding) or had never heard about it, and he often had parents reaching out to inquire about the coding club after it was canceled as a result of COVID-19. Despite the club's closure, he still aims to provide computing learning experiences to his students by offering CS during breakfast, and by integrating CT into his curriculum.

As an elementary school teacher, Mr. Ozo is very familiar with integration. The school schedule contains the same number of hours, but the curriculum continues to grow, and/or new state standards are added. The new or expanded curriculum needs to

get integrated because there is not enough time to dedicate to it. Mr. Ozo finds that "integration" is often used as a magical phrase in elementary education for everything, and CT/CS are no different. He sums it up when he shares, "They [leadership/the state] define the content, but not the context" (first interview). There is no professional development provided on integration and the flexibility feels too great at times.

One of the challenges Mr. Ozo encountered in Minnesota is that the content is not defined when it comes to CT/CS because there are no standalone CS standards. The way CS standards are being added is by integrating them into existing academic standards, but Mr. Ozo shared how that felt it was more like "burying them." Without context and training on CS, teachers may not quite grasp what it means to integrate them or not feel confident in their ability to teach it beyond a nod to the standard. Mr. Ozo articulates it well when he shares that he thinks, "[In] a lot of teacher programs we're trained to incorporate not integrate" (second interview). An example he shares is of a specialist or coach coming in to teach a lesson and then pointing out how they incorporated all of these standards, but he notes that "just because I list the standard doesn't mean that it got integrated" (second interview).

Since he is aware of the difficulties integrating brings and the intentionality it requires, he approaches integrating CT as purposefully weaving it in. Even if it's just a small or basic connection, he works to embed CT into the content:

> integrating means it might be pretty basic, it might be I'm just pointing out those existing connections to CT, and you're not gonna be posting this on you know the district website, no one's going to be patting you on the back. No one's gonna

probably ask you much about it, but you start to build a culture. (second interview)

Ultimately, though, he finds it a balancing act to fit everything he would like into the school day, so he tries to make the most of "extra" time to focus on coding and programming (e.g., breakfast or recess). Embracing a culture of learning and trying new things to improve upon his integration of CT, Mr. Ozo continues to seek out his own (in)formal professional learning experiences and resources such as conferences, participating in professional teacher organizations, Twitter chats, equity summits, and connecting with others who also are working to integrate CT/CS.

The Enacted Curriculum of Mr. Ozo

In order to better understand the enacted curriculum of integrating computing in Mr. Ozo's classroom, TPACK-CT categories are used as a guiding and organizational lens. How he conceptualizes CT knowledge in the classroom (CK-CT), what pedagogies he enacts when teaching CT (PK-CT), and what technologies he uses for teaching CT (TK-CT) are further expanded upon. These categories and findings overlap and intersect with each other because teaching and integration is very nuanced and complex - but what follows is a more vivid picture of how Mr. Ozo works to integrate CT into his classroom, or the enacted curriculum.

Mr. Ozo's TPACK-CT

Throughout our time together it was clear that Mr. Ozo felt empowered to integrate CT due to multifaceted reasons such as his confidence in his abilities to teach CT, his reflection on the visible student successes from integrating CT, the supportive

leadership and "permission" that the academic content standards give, and his past learning experiences. His classroom is a place where "thinking aloud" and trying again are part of the culture, utilizing technology for creating is encouraged, and connections are embedded and revisited frequently. Since Mr. Ozo aims to look at what the students are actually doing and learning (rather than the flashy technology or exciting technology *he* might be interested in), students leave his classroom knowing how CT and programming can be used to promote efficiency, solve problems, and with knowledge of future computing pathways.

Not only does Mr. Ozo make connections between what students are learning and future career paths, but he also exposes them to the computing pathways available at the middle school for students - something they may relate more to as fourth graders. To help with this, he brought in the middle school teacher to get the students excited about what awaits them after elementary school: coding, apps, robots, drones, and video games were some of the highlights. And Mr. Ozo feels like it is working as the middle school teacher has shared that his once mostly boy-filled class now is almost half made up of girls and he has a Girls Who Code club up and running.

The professional development experiences Mr. Ozo has had, in addition to his own educational experiences as a student, are a large part of what have shaped his TPACK-CT. While he grapples with imposter syndrome, he continues to work to integrate CT into his content and ever-evolving curriculum load because he sees something as better than nothing. Mr. Ozo not only works to integrate CT, but also intentionally shares what he is doing with his colleagues because he views CT as a

critical skill for students to have to navigate their futures and a way to illuminate and center equity in the classroom. “I think if my kids are going to do well, they need to have computational thinking to survive” (second interview).

Mr. Ozo’s colleagues play a critical role in his TPACK-CT as they expanded the boundaries of his classroom from not just the room he taught in and his students, but to the times he met with colleagues and collaborated during their shared planning. He shared that, “I think the [CT] pre-lesson planning comes when I plan with my team and it's shifted from ‘where can I get this to integrate’ to more of ‘how can I get [my team] to like it and be intrigued by it” (second interview). He recognizes that students’ exposure to coding is “luck of the draw” and dependent on their access to those learning experiences, so he is motivated to expand on where those opportunities exist by starting with the teachers in his school that are right next door teaching fourth grade, too, - his neighbors. While he wishes they could all participate in the same professional learning cohort that he did because it feels like the missing piece for getting them interested, he still tries to share what he’s doing in his classroom or explain why his students are excited about something. He mentions also trying to “sneak it in” with his colleagues where he will share a CT strategy when teachers ask for ideas to help students - he just will not explicitly call it a CT strategy.

Despite his years of experience, training, background, and student success stories, Mr. Ozo still wrestles with wondering if he is doing good enough or comparing what he is doing to others, or as he called it in one interview, “imposter syndrome.” He has shared that there are times he does not feel like he has knowledge of the things that kids want in

CS (like apps and robotics), that his learning opportunities are not as "cool" as what kids want now. While these fears and worries are very real, they do not stop Mr. Ozo from trying. At the end of the year when Mr. Ozo was preparing for a conference presentation, he came to the realization that he had integrated more than he originally acknowledged. During his preparation, he came across a model that discussed three levels of integration: "exist, enhance, and extend" (Waterman et al., 2020). By applying this model to his integration of CT, he realized how much more he integrated CT - instead of thinking "integration" only counted if he used computational thinking to enhance or extend content lessons, he was able to see all the points where he acknowledged or emphasized the existence of CT in what he was already teaching, which falls under the "exist" level in this model of integration.

Using CT as a lens for viewing other content and applying this "exist, enhance, and extend" model (Waterman et al., 2020) also highlighted for him the existing connections of CT in multiple areas like the English Language Arts (ELA) speaking and listening standards. One of the connections he called out was how sentence stems and conversation practice overlap with some of the CT activities students engaged in as they practiced thinking aloud and using algorithmic thinking to craft sentences. He also felt that using CT as a lens for other curriculum made it easier for him to embrace a new science curriculum the school brought in because he saw the existing connections with CT. He summarizes this when he shares, "So I think that computer science and computational thinking - if you integrate those in elementary that helps you sort of open up how you can integrate anything in elementary…" (second interview).

Conceptualization of Computational Thinking [CK-CT]

As a result of his educational and career backgrounds, as well as his experience and professional development, Mr. Ozo sees integrating computational thinking as the goal because it is multi- and transdisciplinary. He not only identifies the four areas of CT (abstraction, decomposition, algorithms, and pattern recognition), but he also conceptualizes CT as a valuable and essential problem-solving strategy that results in a mindset and set of practices that can be applied in all areas: "[Computational thinking is] the mindset and mental skills required to solve problems" (survey response). He believes CT provides soft skills that are essential for students' success and because of its importance, it belongs in elementary classrooms. He continues to highlight and provide opportunities for students to practice these soft skills while also practicing decomposition, pattern recognition, algorithms, debugging, and abstraction because he has seen the positive impact integrating CT has had on students (e.g., performing better on word problems and elementary content assessments, increased confidence in one's abilities, practice working with ambiguity, students who may be overlooked having an opportunity to shine or mentor, etc).

While Mr. Ozo wrestles with imposter syndrome at times, he still feels confident in his own CS abilities and his ability to make connections between CT/CS to the real world and other subjects - a confidence that appears reinforced by continued learning experiences and supportive networks/connections. Yet he noted that both curriculum to integrate CT in a way that enhances existing elementary education curriculum and pre-developed CT lessons are lacking. A lot of his integration of CT is pointing out the

existing connections to the math and science practices/standards. He shared some examples of the overlaps he sees with the math practices his curriculum uses like understanding problems (decomposition), persevering to solve them (debugging), and looking for a structure to make use of in solving the problem (pattern recognition). Noticing these overlaps, not only made him feel like a better adopter of the math practices, but it also allowed him to see what CT areas his students may need more support in, also helping the students in other content areas, too. He noted that his students are good with pattern recognition, but need a safe place to fail and practice it. He also noticed his students struggle with decomposition compared to other elements of CT, and that abstraction is more easily understood but he feels that the CT connection needs to be added more. "Practicing" CT concepts also involves more than sitting at a desk, memorizing terms, and doing a worksheet. Rather, Mr. Ozo often has his students participate in "games" where they are practicing these skills without explicitly calling it "computational thinking" (see Figure 17).

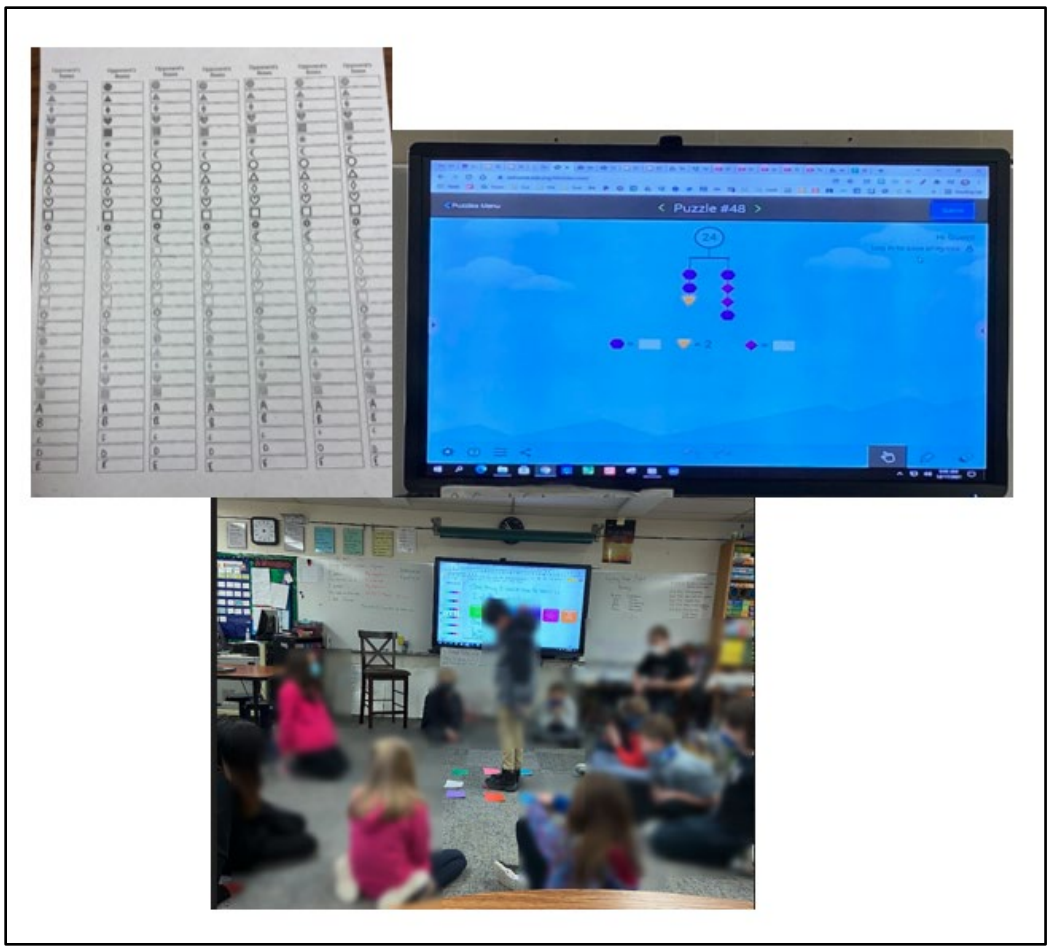


Figure 17. Game and puzzle examples used by Mr. Ozo. He uses games and puzzles Mr. that require using CT concepts and skills, such as number hunt worksheets with partners, a math puzzle, and a "secret path" game.

He shared that while he sees overlaps between computer science and computational thinking, he conceptualizes them differently and finds it more difficult to integrate CS compared to CT. As far as integrating CT, though, he believes that it's possible for all teachers to do. He shared,

> I think that the plus and minus of computational thinking and coding and computer science is that it doesn't have a spot on the master schedule, so if it's

> going…to get in [to the curriculum,] it's going to have to come in genuinely integrated like… weaving. (second interview)

He believes his teacher colleagues are either already doing CT and that it is just a matter of naming it and/or making the connections for their students, or if they are not integrating it, they easily could.

One concern Mr. Ozo has with embedding CS standards into academic standards and not providing training is that within CT there are some vocabulary nuances between CT and other disciplines (e.g., "compute" in math vs. "compute" in CS). He worries that a math teacher might be unfamiliar with the nuanced nature of the differences, but believes PD could help alleviate this as it aids teachers to "make sense" of what they need to be doing or what they have been trying to do when it comes to integrating CT. Ultimately, though, he believes all teachers can and need to integrate CT - "the kids are owed that much" (second interview).

Not only does Mr. Ozo see the relevance and impact of learning CT in his own education just as he tries to highlight connections for students, but he believes that 100% of his students are capable of engaging in CT and computer programming. Since embedding learning and practicing CT into his classroom, Mr. Ozo felt his students could better talk about solving math problems, grasp and solve math word problems, develop more sophisticated science models, and apply debugging skills across content areas. Mr. Ozo sees CT as embedded in his curriculum, an anchor point for teaching fourth grade curriculum, and as a way to develop soft skills that students will lean on throughout their academic and professional journeys. He shared during our second interview: "There's a

2025 World Economic Forum research, it talks about the skills you need - it's all the soft skills, it's basically elementary class, as well as computational thinking plus soft skills" (Whiting, 2020).

Rather than creating a paper assessment or one that may feel "academic" to measure students' understanding/learning of CT, Mr. Ozo looks for lasting impacts:

> I don't have like a paper assessment…but I do think that you know if my colleague [who teaches at the middle school] is telling me the number of girls is going up [in his computer science classes], and he now has a girls [coding] team - those are good assessments.

He also adds that seeing how students transfer their computational thinking knowledge, something he conceptualizes as interdisciplinary, to other content areas is another way to assess CT knowledge:

> ...one way I can kind of assess [computational thinking], [is] by what am I able to do with them [in other content areas]…For example, if we were doing fractions: Can they do fraction word problems? And are they…doing the modeling? Are they able to decompose numbers? Are they able to break it up into steps? All that stuff...you can't do word problems unless you have those done. (second interview)

Pedagogies for Teaching CT [PK-CT]

Using connections, fun, disruption/deconstruction, and culture as key pedagogical strategies emerge in Mr. Ozo's pedagogical content knowledge. Mr. Ozo has made practicing CT a part of the classroom culture and feels that he is effective in teaching and integrating CT, which is evident in the many overlaps and connections he often makes

between CT and other content areas (e.g., math, science, and ELA). He sees CT as something that is woven into everything; he uses CT as both a vehicle for content and a strategy to also "churn up" his classroom and disrupt students' misconceptions and established beliefs they may have about themselves. For example, he encourages students to not only be social and collaborate, but to also mentor and help one another. He will intentionally have those that may not normally see themselves as "being good at math" assist those who may be considered higher achievers in math when doing CS activities. He has created a culture in his classroom where debugging is normalized and encouraged; he provides a safe space where students can make mistakes so that they can practice trying it again.

Mr. Ozo's classroom is full of movement, chatter, anecdotes/connections, and collaborative work - strategies that he applies to all content areas, including when teaching and integrating CT. Some of the specific strategies he uses that both integrate CT and teach the content are activities like sentence stems, modeling activities in math and science, number talks, pattern recognition games, videos to emphasize connections, word problems, creative and open-ended projects, unplugged activities, and inviting others outside of the classroom to come in to share their experience while simultaneously embedding connections beyond 4th grade. During one of my observations, students were sharing sentence stems (while decomposing the parts of a sentence as they practiced) related to what they wanted to learn that day and answering a technology question, "What app would you want to be created?"

One of the strategies Mr. Ozo has both learned as he has participated in professional development and volunteer experiences around CS, and that he also employs in his classroom, is the importance of letting students play and have fun. While he emphasizes the fun of CT (something he learned and experienced in his own PD encounters), he finds that balancing fun and structure is an art that he is still learning, particularly when it comes to technology. He recognizes that teachers can often get stuck in the mindset of "rules" and "this is how it's done," but students need to be able to play and tinker - especially in their initial introduction to computer science. To help provide structure and options, he tries to include video introductions to the technology students will be using (in small chunks and often during breakfast) and then models the steps from the video for them, and/or he might play a video in the middle of work time with ideas of how students can enhance their projects.

Since Mr. Ozo notices that his students often struggle with connections in general (e.g., remembering a lesson from the previous day, transferring strategies to other content, etc.), he works to disrupt this by repeatedly emphasizing the importance of connections. He points out,

> They [elementary education students] don't connect to the previous day, they don't connect cross-curricular…I think part of that is how the elementary day is structured, I think we all, the kids almost [are] just like 'next, next.' We have to go back and kind of deconstruct that I think. You can float through the elementary machine and get good feedback and not have any clue why you did anything. (second interview)

He aims to deconstruct that mentality through purposeful, explicit, and relevant connections between CT, the content they are learning (or have learned), and the things outside of the classroom. During one of my observations, after having students do a number hunt activity (an activity that implicitly involves decomposition, pattern recognition, and algorithms), he had students come together where he shared an example of a spreadsheet he would use as an engineer (see Figure 18). He explained that no one would want to go through the thousands of rows of data line by line, but instead they would want to find a system in order to quickly find whatever the person is looking for in the data - similar to what the students are doing when they play this game. He tries to view things through a student lens so that he can be intentional about what and who he brings in for students to make connections beyond the immediate lesson and the fourth grade classroom.

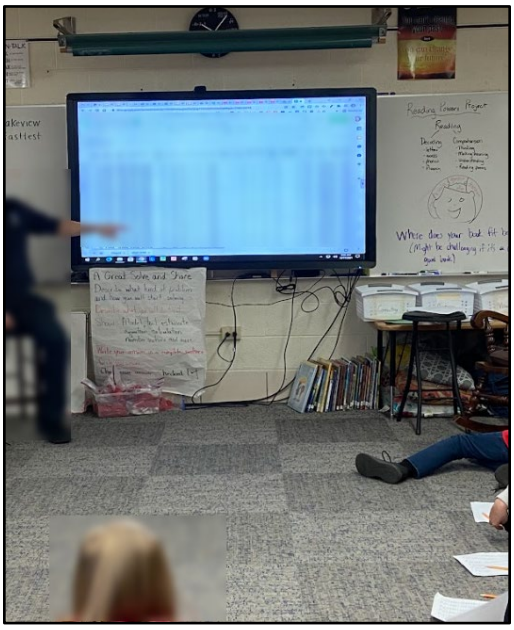


Figure 18. Mr. Ozo shares a spreadsheet of data he would have used as an engineer.

Skype a Scientist (www.skypeascientist.com), a free resource where scientists volunteer to skype into a classroom, is an example of something he brings into his classroom to help students make connections.

> If we Skype with a scientist, then you have to be talking about it before and after so it integrates with like 10 days of stuff…I think we can leverage more if we think intentionally like, 'Okay, how do I get my kids to connect to this?' (second interview)

In addition, since he has done about two Skype a Scientist activities each year for the last ten years, he has about twenty different fields and stories he has collected that he can pull from to help embed connections between CT and beyond fourth grade for his students. He brings in other examples and connections as he relates to his students and shares his own stories with them of how things have changed over time to showcase how prevalent technology and CT are across many fields. An example he shared with them was how his grandpa was an electrician and how much that field has changed. He explained the role of technology and apps that linemen now need to know when they access power lines while they are working out in the field. He brings things to life by connecting CT to life beyond the classroom with numerous stories and examples.

Integrating CT and CS into his curriculum and classroom, have also allowed Mr. Ozo to disrupt the "traditional" classroom hierarchy where some students can become known to be higher achievers than others or to build students' own self-efficacy: "...every kid is getting experiences struggling and debugging and being successful and having to share and having a mentor. And that is hard to do without the computer science piece"

(second interview). He has seen how CT can be an outlet for often underrepresented students and wants to provide as many opportunities as possible for students to feel success - something he feels that CS can provide.

Students spend a lot of time talking about their thought processes, what does(not) work and why, and learning how to solve problems in Mr. Ozo's classroom. During my time in his classroom, I saw students practicing this skill throughout a myriad of subject areas - they were encouraged to explain their reasoning and to listen to each other. Students did not just answer and get told they were right or wrong and the day moved along, but time was spent on understanding students' thought processes so that they could also get to the right answer - of which there were often multiple paths to get there and possible answers. Being in his classroom felt like being in a place where "failure" was normalized and students did not appear afraid or embarrassed to share why they approached a problem in a certain way - it was truly a part of the learning experience and culture. In his lesson reflection he shares that he wants students to:

> have experience with productive struggle…students need to know that interesting problems are often very hard and take much longer than they envision to solve…As students shift in 4th grade to multiple step problems that require multiple strategies…students start to associate normal problem-solving challenges as deal breakers and that they aren't 'smart enough' for a particular problem, or worst case, an entire subject. I want students to see that struggle, mistakes are normal, and to make those common experiences.

He explains that this is a shift in his classroom culture that can be attributed to intentionally integrating CT practices, like breaking a problem down into smaller bits (decomposition) or being willing to try again when the first solution does not work (debugging).

He can see these shifts across multiple content areas: students are more apt to create their own science model rather than just copy down the one he shares, students can more sophisticatedly explain their science models and solve math word problems than prior years/their peers from other classes, and both he and the students were more easily able to adopt a new, more open-ended science curriculum. He explains,

> I think the computational thinking helps because I, to be honest, like about three-four years ago, I would have had students freaking out [about the new, less prescriptive science curriculum] because, 'like, but what am I [the student] supposed to do?'... but [this is] a new culture for these guys because before it was, 'you have to draw what the teacher put up on the board and show you remember it…' So I think the CT, especially, is more like: no, can you figure it out. (second interview)

The students also recognize these things as they shared what they thought of computer science in a SeeSaw activity Mr. Ozo organized, many of them pointing out the value they find in it, including one student who shared, "One skill that I'm learning is that when I'm coding, I won't always get it the first time." He and the students have embraced a culture of trying new things and figuring it out.

Technologies for CT (TK-CT)

Technology not being essential nor a requirement for integrating computing emerged as a piece of Mr. Ozo's TPACK-CT. As an elementary educator working in a small, more rural elementary school that does not receive large amounts of funding, it makes sense that Mr. Ozo both believes and practices that computers are an asset, but not required to integrate CT. Rather, he sees technology as a way to highlight the power of CT and CS for students, and as a way to connect to students' interests. He uses Chromebooks, iPads, robots (e.g., Ozobots and Spheros), and online resources or apps (e.g., Google CS First, Code.org, Skype) to expand students' access to know what's out there, to practice coding, to see the automation of problem-solving, and to get students engaged.

When incorporating technology into his lessons, Mr. Ozo encourages students to play, talk, and share – he knows that this is critical for students to buy-in and get excited. While he enjoys bringing in robots and exciting technology, he feels like he sometimes runs out of time or is not as prepared to teach the coding side of it. He has also found that in the past he might have a challenging project idea, but if it lacks a purpose or does not feel beneficial, he has noticed students (particularly girls) disengage. Ultimately, as he tries to find a balance between play and structure, he wants to ensure his students are using the technology to create (rather than consume) and to solve problems that they find a purpose in.

While exposing students to technology and programming, he works to build up their programming skills and expand their conception of who can code (similar to how

his own preconceptions were disrupted). He shares that, “While you can teach ‘unplugged’ computational thinking activities, it’s using technology to solve problems that helps students see the power of this skill.” One lesson he taught during observations that was a blend of unplugged and technology was on fractions. After students participated in a number talk around fractions where students came up with an answer to a question related to fractions, talked through and shared their reasoning with each other, they then came back together as a class to discuss what they learned and remembered about fractions. Then they brought up the fractal program coded in Scratch (see Figure 19). Mr. Ozo had them “be the computers” in order to explain and physically write out/practice what the blocks were doing to solve the math problem. After they did the work by hand, they were able to test their answers by running the Scratch program. If the answers didn’t match, Mr. Ozo walked them through debugging to find out why. He pointed out that getting to play a game you just made is part of the reward of coding and that students are often motivated by having fun/playing. This fraction lesson aided them in understanding how the coding worked. Mr. Ozo uses technology to make CT come alive for students and to engage them, but he does not solely depend on it to teach CT to students.

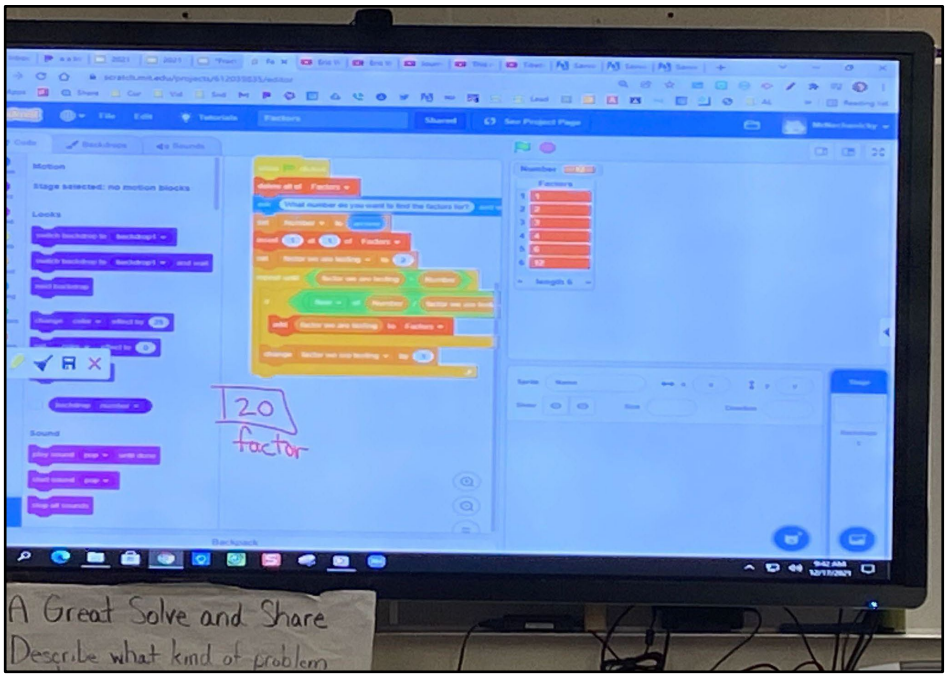


Figure 19. The code behind a fractal program in Scratch. Mr. Ozo has students look at the code as they practice being the computer to solve math problems before using the program.

The other value he finds in bringing in technology is that it makes future CS pathways more visible to students – from future middle school classes to possible college degrees and career options. In fact, he really enjoys that students articulate more specific, diverse careers that they "want to be" as he's worked in more connections for them, such as oceanographer, app developer, and system engineer. Digital resources are often used to expand students' exposure to computing pathways and to help them make connections between CT and beyond the fourth grade classroom. Examples of this include how he concludes the students' day with a 2 minute video where they identify the four domains of CT and utilizes the Skype a Scientist resource.

Summary

Mr. Ozo has found that integration provides freedom and can result in so many connections, but it can also be challenging. He shared that "integration gets inhibited" for teachers by fear of being questioned on academic standards. Not having standalone CS/CT standards means that not all students are receiving quality learning experiences, not all teachers are integrating it or even fully understanding it, and it can lead teachers to judge their own enacted curriculum because there is no intended one (standards) to compare it to. He shared that he feels comfortable pausing certain curriculum to prioritize others, and that he is willing to sign up for trainings and figure it out; however, he knows that not all of his colleagues feel that way. He sees that there's a fear of "getting in trouble" when deviating from the prescribed curriculum, or teachers have had prior leadership that micromanaged them so they're afraid to not have permission, and/or that some teachers worry about not being the expert on a topic they're teaching. He points out that a teacher has to feel comfortable and supported to actually integrate because sometimes it might mean deviating from the mandated or bought (intended) curriculum in order to fit everything in. Networking and getting teachers together to see other ideas for how things can look are powerful ways to deconstruct some of the fears that teachers have when thinking about true integration.

Collaboration is something Mr. Ozo values for both his students and himself. It seems to be a critical piece in disrupting beliefs about oneself and abilities - both for the students and the teacher. Not just allowing, but encouraging students to collaborate and help each other, provides a time for students to shine and work through mistakes together.

Talking to others at conferences and on Twitter helped Mr. Ozo acknowledge all that he was doing and disrupted some of the imposter syndrome that can creep in, as well as expanded ideas for CT integration lessons and connections.

Mr. Ozo's passion and dedication to not only integrating computational thinking, but to teaching, shone through during our interviews where his answers and anecdotes often drew upon an article, conversation, Twitter exchange, conference, research, or presentation that connected to the conversation. Not only did he emphasize the importance of helping students make connections beyond fourth grade and the immediate content via computational thinking, but he practices it in his learning and connections within his own teaching practice.

Equity is both explicitly and implicitly woven into Mr. Ozo's practices and classroom. One of the motivating factors for him to originally bring in a coding club was hearing how the middle school teacher had an almost all-boy class. He also noticed that students who were often overlooked in the mainstream classroom actually excelled and thrived in the school's makerspace. He desires that all of his students feel successful and have multiple venues/outlets where they feel good about themselves; he wants to disrupt those preconceived beliefs about what a student can(not) do. He shared,

> I think that if you start to learn computational thinking, you have to start noticing that the kids don't perform the way you predicted, which is an equity issue. If you start predicting who does well, that's the problem right? (second interview)

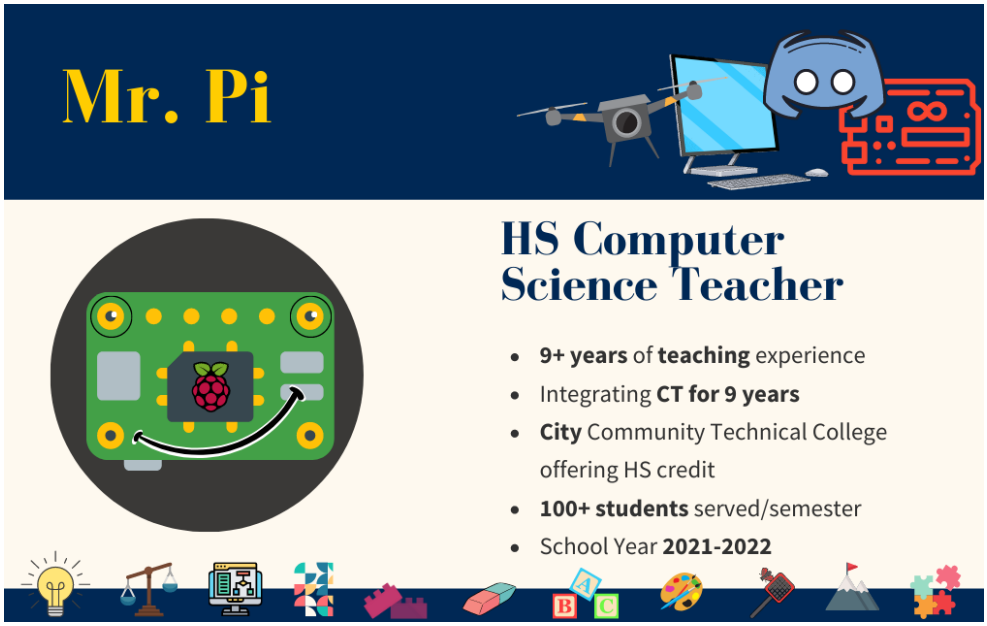
He also noted that his learning disabled (LD) students are "often used to listening extra hard to directions" and have a mindset of "I'm going to have to probably try several

things before this works," and so these students tend to pick up CS quickly. They experience feeling successful as they often help those who might be considered "high flyers" in math or other content areas. He is familiar with research on the importance of getting students hooked into CS by middle school if there's hope of them seeing themselves as computer scientists, so creating those opportunities are something he wants all teachers to provide their students.

"If I look at what my kids need in this room, I think they need what computer science is bringing in" (second interview). Mr. Ozo's integration of CT changed not only his classroom culture, but also the computing pathways for his students. The middle school teacher can no longer rely on just Scratch, his classes are more balanced, and he has had to expand club offerings. The shifts Mr. Ozo has made not only in his classroom, but also in his teaching practices since integrating CT, were evident as he discussed the positive impacts on students and shared specific stories of students. Mr. Ozo's advice for teachers who want to bring in (more) computational thinking is to just start with existing connections, then tinker with coding, and be ready for when there is more "enhancing" curriculum available out there.

> I would tell people [to] give yourself a break; there is not an easy way to enhance math, science, reading [with CT] - there's little tidbits out there. But then I would tell them like if you started now that's kind of cool because you have all these existing connections that no one really says are allowed (and by allowed I mean they're not valued). And I think it's going to come the next 5-10 years, so if you don't start now you've missed the boat, so I would say, **start**. (second interview).

CHAPTER 6: MR. PI



"*I feel very lucky in this district, actually, that I have the freedom to do what I'm doing.*" - Mr. Pi

Figure 20. Mr. Pi's introductory ID card.

Introducing Mr. Pi

Mr. Pi taught computer science courses for high school students at the local community technical college (CTC) during the time of this study (2021-2022). The CTC provides college credit-earning courses for high school students. Mr. Pi had been teaching a combination of math and CS for about nine years when this study began. His background is in computer science and his foray into teaching was when his child's private school urgently needed a math teacher. They knew him as a "math guy" since he helped out with their school website and hoped he could teach some math and technology

classes at the school. After he found teaching to be fulfilling, he completed his master's in education to teach math at the secondary level.

He began teaching at public schools seven years prior to the study and was eventually asked to create computing pathways for a new high school that was planned in his mid-size city. Due to budget cuts, the plan was reimagined into creating computing pathways at the town's local CTC where students from local high schools could participate. Introductory courses are offered at the three high schools in town, as well as advanced placement (AP) computer science courses, and then the more advanced/specialized CS courses are offered at the CTC. Up until the 2021-2022 school year, Mr. Pi taught at all of the high schools and at the CTC concurrently, but now each school has someone to teach introductory and AP CS classes so he works solely at the CTC.

The city where the CTC sits is one of the largest in the state with a population over 100,000; one of its defining characteristics has historically been different large companies or industries bringing in jobs and workers aligned with the industry. The current dominant industry in the city brings in people from throughout the world. However, Mr. Pi notes that despite the typical Midwestern town feel that there are some disparities and disconnects between those who come to the town for a job in the main industry versus those who live or migrated there, particularly in socioeconomic statuses. Since the town is rather large, the CTC, where Mr. Pi's classes are offered, worked with over forty-five high schools and had over seven hundred students attend the high school college credit program each semester of the 2021-2022 school year. The public school

district in the town is one of the largest in the state with over 25 schools in the district. The three high schools served 5,100 students during the 2021-2022 with the following combined demographics: 57.7% white, 14% Black, 11% Latino/a, 10% Asian, 6% two or more races, and less than 1% American Indian/Alaska Native (U.S. Department of Education). Mr. Pi taught semester-long sections of multiple advanced computing courses (e.g., courses focused on Python, web design, data structure, and IT infrastructure) that lasted a little over an hour every day, condensing a year-long curriculum into one semester.

Computing and Classroom Context

When Mr. Pi taught the introductory CS courses at the high school, he found that there was not only a variety of students trying the course out but it also appeared to be desired by students as there are now three sections of this course at every school, in addition to other CS courses offered at the high schools and at the CTC. The introductory courses were specifically designed to require no prior experience and as a way to introduce students to different technology each week with the hope that they would find something they are interested in. In comparison, the more upper-level courses that Mr. Pi now teaches at CTC are usually taken by students that already have an interest in CS and the goal is to keep them interested; most of the CTC CS courses also require a prerequisite. Despite the large number of CS offerings and teachers needed to teach them, all of the CS teachers across the schools in the district (including Mr. Pi) only meet once a year during the summer to review things. There is not a coordinated curriculum but they

try to stay on the same page/remain aware of what is happening across the schools, yet Mr. Pi feels there is limited collaboration.

During the 2021-2022 school year. Mr. Pi offered semester-long classes focused on data science, JavaScript, robotics, Information Technology (IT) infrastructure, and algorithms and data structures at the CTC. Enrolled students were bussed in every day and the courses are block classes, meaning that they last for two periods (a little over an hour). His classroom has long tables with computer workstations, each row having around 4-7 computers that students can sit at. The rows face the front of the room where Mr. Pi's desk and computer sit beside the projector screen (see Figure 21). Cabinets and shelves with supplies, equipment, and electronics line the perimeter of the classroom.

Figure 21. Mr. Pi's classroom arrangement.

During my observations there were about 15 students in each of the sections, although some were missing in the physical space due to conflicts (e.g., band practice, out sick, etc.). Students in the courses I observed were comfortable walking in, going to their computers, listening to their music, and getting started for the day. A small number of girls (1-3) were in each of the classes I observed, oftentimes seen sitting beside each

other if there was more than one. Mr. Pi banters with his students both in person during his classes, as well as online using their class-designated Discord server (a free communication app, https://discord.com) - allowing students to choose how and when they participate, both in real time and asynchronously.

Mr. Pi expressed his appreciation for his administrator and the freedom his current role gives him - "I don't have to fight quite as many bureaucratic battles as I might [in a traditional public school setting]" (first interview). In addition, because computer science is such a specialty area, he feels that he has less fear of school leadership coming in and telling him what to do because they are not as familiar with it and there are no required state CS standards to adhere to. However, he notes that for many teachers fear is a huge barrier in teaching and/or integrating CT/CS as there are false beliefs to overcome ("I'm not a math person" or "tech person") and teachers often do not have a background in the subject so they don't feel like an expert on it. He explained that he sees elementary education as the place to integrate because they are more willing to "try stuff like Bee Bots" whereas in the older grades,

> Teachers are more and more put into a box into what they're supposed to teach and [to] convince them to…either give up some time for tech or trying to convince them that actually this…integrating it [CS/CT] makes it better. That's a hard sell… (first interview)

He sees the importance of having support and community, especially for teachers who are new to integrating CT or CS into their classrooms, as they work to "get over that hump [of what] CT is" (first interview). Since schools are often not able to or not willing to

spend money on technology integrationists to offer this support, Mr. Pi finds professional development one resource in developing community and support for teachers teaching and/or integrating CS and CT.

Not only does Mr. Pi himself offer professional development opportunities for any interested teachers in his district to learn about computer science via a "choose your own adventure" option, but he has also been a participant of Project Lead The Way (PLTW) (pltw.org) and Mobile Computer Science Pathways (CSP) (mobile-csp.org) professional developments. He especially appreciated the intentionality about inclusion and equity in the Mobile CSP trainings and the supportive community that resulted. He feels this was a good PD for somebody new(er) to teaching CS as it gave teachers a voice while still providing more personal support, or "hand holding." In comparison PLTW felt a little more prescribed for him as he felt they were "trying to do everything a little bit, but nothing really well." However, one of the take-aways for him was the instructional design approach they go over that scaffolds student learning: an activity-, project-, problem- approach (APB). In addition to these two PDs, he has stayed involved with professional teacher associations, exploring resources on his own, and pursuing a postgraduate degree - all of which inform his own practices and provide community.

For Mr. Pi, he sees intersections with CT and other subjects like English and social studies and continues to get ideas from combining lessons and resources. Despite his depth of knowledge and the intersections he sees, it still takes effort, time, and experience to make relevant connections between what students are learning and beyond the classroom, as well as to encourage more discussion around the problem-solving

strategies (CT) they are employing in computer science. For example, he is still working toward the goal of making his Python data science course more social justice-oriented in the future.

While he appreciates the flexibility of his current role because he is associated with CTC and not with public school, he does see having no state requirements as a double-edged sword. One of the challenges he shared is that (math) teachers are often "voluntold" to teach the computer science courses because Minnesota does not have a CS licensure endorsement/path. Since the course becomes an added responsibility, Mr. Pi thinks teachers may not be interested in pursuing more learning opportunities and/or might feel insecure in their expertise of the subject so they may not be as willing to engage in collaboration. In addition, usually the first course they are asked to teach is an advanced placement (AP) CS course because it "looks good" and has a prescribed curriculum. "So I think what happens is teachers who are sort of shoehorned into teaching CS, especially if that's their first course [AP CS A] think that that's what CS is and what it should be like" (second interview). As someone who had success in their computer science education and in the industry, he knows how easy it can be for people to teach based on how they were successful. He finds that this happens a lot with teachers who are first introduced to CS instruction via AP CS courses, which have a more esoteric nature and have a connotation with "success." Since Mr. Pi has strong feelings about AP CS courses ("[They are] designed to extract any love of computer science from the students taking it and replace it with misery" (second interview)), teaching all CS courses in that manner does not align with Mr. Pi's feelings that CS should be integrated into

curriculum rather than schools just offering CS. He wants students to see the meaningful, authentic tasks that they can use CS for. He recognizes that if teachers are teaching all CS courses like AP CS A or if schools are only offering AP courses then there is probably not a lot of transference happening in many high school computer science courses. He explains,

> ...I think we're a long way from making the focus [fun/transference] and not just sort of 'hey here's a cool project,' but like we need to design curricula with this [fun, engagement, & authenticity] at its core…the authenticity of the project is important, and not just a cool slideshow. (second interview)

Mr. Pi had the advantage of having experience in the CS industry where he encountered many different ways CS was used and was exposed to diverse technology, and as a result he felt comfortable bringing those things into the classroom because wanting to share the fun and engagement in CS was a part of what initially fueled Mr. Pi's desire to teach CS.

Mr. Pi shared that originally his motive for teaching CS was based on his own experiences of enjoying computers and wanting to share that love of CS with students to hopefully also spark joy for them. After teaching in public schools, though, his awareness and exposure to less privileged youth and experiences that differed from his own shifted his motivation. He feels he now has a moral obligation to provide learning opportunities of how to use CS tools to empower people who have been historically marginalized and oppressed. He also states that, "Computer science is the most positivistic, high barrier, systematically racist subject that currently exists" (second interview). With this in mind, he approaches his work with an equity lens. One example is his data science course that

he hopes to build more transference into so that students can see how they can use CS to analyze larger topics that they find relevant and engage in discussions about what they discover. His postgraduate studies and research interests have also impacted his approach to computing:

> And, you know, that is my area of interest in research is changing the culture of computer science to value those communal goals and communal change, and at least equally, if not more importantly, than agentic sort of 'let's make an app that makes me a million dollars' kind of goals. (second interview)

The freedom and support Mr. Pi experiences in his role at the CTC, his passion for learning, and the role of computer science in his own life are visible as he notes the growth of CS classes in his community, reflected on his teaching experience through the lens of equity, research, and courses he's experienced in his current pursuit of a postgraduate degree, and shared how his reasons for teaching CS have shifted over time. These experiences and beliefs all intertwine to inform Mr. Pi's enacted curriculum of teaching high school computer science.

The Enacted Curriculum of Mr. Pi

Since Mr. Pi teaches the subject of computer science, CS is often focused on more than CT. While CT may not be as explicit in the findings from the analysis of the data, the TPACK-CT categories still help organize and articulate what Mr. Pi's TPACK-CT looks like when enacted in the classroom because CT is often inherent within CS. How he conceptualizes both CS and CT knowledge in the classroom (CK-CT), what pedagogies he enacts when teaching CS and CT (PK-CT), and what technologies he uses

for teaching CS and CT (TK-CT) are further expanded upon. These categories and findings overlap and intersect with each other because teaching and integration is very nuanced and complex - but what follows is a more vivid picture of how Mr. Pi works to teach CS and integrate CT into his classroom, or the enacted curriculum.

Mr. Pi's TPACK-CT

Supportive administration, years of experience and success in the CS field, collaborating with other teachers who are doing exciting things in their CS classrooms, studying research in his postgraduate degree, and seeing the potential of CS to be taught in a way that empowers students to engage within their communities are some of the key factors that influence Mr. Pi's TPACK-CT. Not only does he enjoy and have fun teaching and participating in computer science, but he places great importance and responsibility on teaching CS in order to empower all students. With an eye on equity, he tries to enhance his courses and teaching practices every year as he desires to provide social, collaborative experiences for his students to enjoy CS, and then challenge them to push and grow their knowledge and understanding while still tasting success and having fun. Over time he hopes to support a shift in CS culture where communal goals are just as, if not more, valued than individual, agentic goals. These experiences and beliefs are evident in his content knowledge, pedagogy, and use of technology, including how these all intersect to create his TPACK-CT.

Conceptualization of Computational Thinking [CK-CT]

Mr. Pi sees computational thinking as problem solving and something that is transferable across situations. He further defines CT as "breaking down a problem into

parts that can be approached individually, being precise with instructions to remove ambiguity, testing individual parts and the whole to see if it solves the problem, revising and re-testing as necessary" (survey response). While he views computational thinking as more similar to mathematical problem solving, but with more creativity thrown in, he does see it as a way to build students' self-efficacy, hook their interest, and build important skills like decomposition and debugging. He believes that CT should be taught in elementary schools and sees it as an important 21st century literacy. However, he does see CT treated more as a catchphrase, a "mythical thing." He emphasizes that CT itself is not something that will lead kids to just solve problems logically or be able to transfer CT onto other subjects: "...so CT is a great idea…I think it is important and we need to be better about using it and transferring it explicitly to other domains or better yet not siloing CS and using it in other domains" (first interview). While he does not think computers need to play a huge role in CT, he does believe that computer programming knowledge is helpful in most careers and that students are using CT when they engage in computer science.

While Mr. Pi struggles with the phrase of computational thinking, he does believe that students use the metacognition of CT to accomplish using tools in an advanced way during his classes. So he may not explicitly call it out or use CT terminology in the more advanced computer science classes, but he does note that it is a part of students' experiences in practicing and learning computer science. Yet he explained that using explicit techniques such as having students create an algorithm to teach someone to make a peanut butter and jelly sandwich feels like a more transferable way of teaching CT,

rather than programming or having students write loops: "...you can frame almost anything in a computational way" (second interview). Ultimately, his emphasis is just that teaching CT and CS should be authentic.

Thanks to his own experiences and education, he is very confident in his own CS and CT abilities, and feels confident in his ability to teach computing lessons, connect them to daily life, and integrate CS into other subjects. While he does not have a set curriculum, he does use a variety of resources and experiences to build the foundation of his different advanced computing courses. He values the creativity, authenticity, and empowerment he feels learning CS and CT can bring, but struggles with AP CS courses often being the only exposure both teachers and students have to CS as the curriculum is too prescribed and esoteric in nature. Instead, he would rather weave in relevant, fun, and practical opportunities for learning CS into his curriculum.

The focus on CT and the resulting take-aways vary depending on the course and its level. For instance, Mr. Pi thinks the introductory courses really work to excite curiosity and promote self-efficacy as students see that they are able to problem solve challenges they encounter in the class. In comparison, he finds the courses he currently teaches to be more about, "how do I use these tools in a more advanced way" (second interview). He points out that CS is often taught as "pure CS" without metadiscussion or explicit transference points expanded upon. He shared the example of how teaching algorithms and data structures feels more like "pure science" where you are studying the tools themselves compared to a class like data science where he feels more connections can be made beyond the classroom.

When asked about his beliefs regarding how many students practice CT or can do programming, he shared that he believes about 10% of all students can do computational thinking because they do it naturally or without realizing that they are doing it, and that while he feels only about 5% of all students have been exposed to enough technical computing to be able to figure out writing a program, he believes that 90% of kids can be taught to do programming and CT; it is a teachable skill (second interview). Despite his belief that 90% of students could be computer scientists (if that is just defined by having an interest that goes beyond the surface level and into how things work; e.g., playing a video game doesn't make you a computer scientist but rather going deeper and figuring out how they work or the computers work is an example of going beyond the surface level), he believes that most students would instead consider themselves programmers rather than computer scientists because they have not been externally validated as computer scientists.

Learning CS is often touted as a way to provide opportunities for students to pursue their individual goals (e.g., building an app and then getting rich from selling it), but Mr. Pi sees its real value in that it's a way for students to help their own communities. Mr. Pi conceptualizes computer science as one of the most powerful and practical tools developed, particularly for those who have historically been marginalized or oppressed. CS is a tool he hopes his students can learn so that they can also use it to aid their own community. This vision for communal empowerment is evident in how he approaches teaching CS and CT.

Pedagogies for Teaching CT [PK-CT]

Repetition, building on concepts, allowing for creativity, utilizing visuals, focusing on portfolios and progress, and sparking student interest are a few of the strategies Mr. Pi uses to create relevant, authentic, and engaging computer science lessons and projects. He also engages with his PD communities, professional organizations, and instructional coaches to get ideas to enhance lessons, collaborate, and be inspired by what others are doing in their classrooms. In his teaching, he aims to connect what students are learning to daily life. While he thinks CS should be integrated across all subjects, he feels it can be more difficult to find points of connection/integration in more advanced computing classes that focus more on the science of CS. Ultimately, he aims for his CS courses to foster community empowerment, student choice, and a place of belonging, whether or not explicit transference examples/connections are always visible or emphasized.

"...Let's have the fun first and then learn why, how it works…once they're already hooked" (first interview) is how Mr. Pi approaches teaching CS to his students. He shares that he really likes to avoid "boring things" and wants his students to have fun. Some of his strategies to engage students include project-based learning, reflection questions "to help students explicitly understand the steps they are going through to solve problems," and creating relevant connections to what is happening in the classroom and beyond. When he brings in projects, he emphasizes that he is not worried about how the project works, but is more interested in what students can show that they learned from it. Something else he encourages his students to do is to push themselves - if they are

finding something easy and boring, he challenges them to try something they don't know how to do. Making advanced classes engaging, like Javascript web design, feels more difficult to Mr. Pi because he feels those classes focus more on studying the tools themselves. Despite this, he does believe data science is one advanced course that can feel engaging, authentic, and communal. Specifically, he has ideas for how it can be equity-focused and illustrative to students of "the power of approaching…problems in a very rigorous way, and…computational thinking way….and using it as a powerful tool" (second interview).

Overall, he is hopeful that the CS curricula available is beginning to grow and bring in interesting ideas and activities. Mr. Pi noted that CS can be more about creation than other subjects so instead of worrying about grades (something he is not a proponent of), he tries to focus on creation and portfolios - "you want to show kids what they can do and how they can do it and let them do what's important to them" (second interview). Not only does he believe CS can be used in other subjects (art, social studies, physics, and math are some of the examples he gives) to create or make things, but his postgraduate studies and the recent research he has encountered have inspired him to find ways to make CS less individualized and more about "fluid synchrony" (Ruvalcaba & Rogoff, 2022). He finds that pair programming and CS have a lot of western influences and tend to be more of a negotiation for power compared to a more Indigenous approach which is more of collaborative, communal approach according to studies he has read, like Ruvalcaba and Rogoff's 2022 paper on "Children's fluid collaboration versus managing individual agenda: Cultural differences in pair programming." Not only does studying

this interest him, but he has seen and is familiar with research on how the current structure of computer science is more focused on agentic goals which often keep women, Black, Indigenous, and students of color out of CS. Having communal goals and incorporating group work are strategies that he finds to be more equitable, even if they may feel hard to implement with certain advanced courses.

Community and creating a sense of belonging are critical pieces of Mr. Pi's pedagogy: "...building a community in the class is the most important thing, right. Much more important than anything else, and making kids feel like they belong, is the most important thing" (second interview). He believes that identity is really about relationships and not the individual, so he wants to make sure that he creates community in the classes he teaches by helping foster relationships for students. The classes I observed had a community-esque feeling as students chatted while working on projects (both in person and online via Discord), music played in the background, laughter was heard, and most students appeared to be engaged with a partner or two. Mr. Pi explained that he is also comfortable trying to change things on the spot if he notices students are disengaging and he will try to notice what's different - why something may have worked in the past but is not working with the current group of students. Ultimately, he just vows to "do [his] best in the moment" (second interview) and then try to improve upon it for next time if possible.

One of the projects Mr. Pi introduced to his Robotics class during an observation was a classwide Rube Goldberg machine. Using a music video, he introduced the concept of a Rube Goldberg machine (a machine designed to perform a task through a set of chain

reactions), as well as the project guidelines. Some of the guidelines he shared helped to "force interactions," which would then lead to collaboration. For example, each person's contraption would need to be connected to two others so students needed to coordinate with each other in order to form a chain. He shared ideas for what students might use (cardboard, Arduinos, motors, power supplies), shouldn't use (liquid, projects, computers), and shared that he will get supplies if any are needed. The rest of the time was given for students to map out their contraption. During this time, the community and collaborative nature of the class was visible as the room filled with chatter, students walking around and getting ideas from each other or checking out available supplies (see Figure 22), asking questions and brainstorming with Mr. Pi, and Mr. Pi occasionally calling out a new rule (e.g., an Arduino must be used) or tossing out ideas. They leave knowing that they can message him for supplies, encouraged to Google things they do not know, and that the intro video will also be available online.

Figure 22. Students and Mr. Pi look at some of the available technologies and supplies in Mr. Pi's classroom.

Student choice and ownership is emphasized in Mr. Pi's classroom as he utilizes a project-based, exploratory approach to teaching computer science. Some scaffolding and

structure is provided for students while working on their projects, but multiple options are usually available to students, allowing them to make decisions and pursue their interests. Mr. Pi works to ensure his classroom is communal and fosters a sense of belonging, two pieces he acknowledges as critical for providing equitable experiences, no matter the content.

Technologies for CT (TK-CT)

While Mr. Pi has a lot of diverse experience, knowledge of, and familiarity with a variety of technology (e.g., Raspberry Pis, Arduinos, Scratch, drones, etc.), he believes that expertise or even familiarity with technology is not required to bring it into the classroom. He asserts that most things can be figured out as you go and that students should also be encouraged to search for answers because that is how life works outside of the classroom, too. Despite Mr. Pi's belief that it is not hard to integrate CS and technology, he does emphasize the value of personal relationships and support for new teachers to help increase their confidence and comfort in using tools they are not familiar with. He recommends using technology that might feel more familiar in combination with new technology to accomplish a CS/CT goal - this is how students can make and create "cool things" while having a more hands-on, tangible experience.

Mr. Pi sees technology as providing a visceral way for students to develop computational thinking skills while also receiving immediate feedback. Using a computer when programming allows for visualization of algorithms and a hands-on way to debug with immediate results. When discussing the role of technology in making students feel successful, he shared, "The tangible joy in success is contagious" (first interview).

Working with technology integrationists/coaches and/or collaborating with other teachers can be helpful in designing ways to authentically bring in technology, particularly for those who may not be familiar with the technology. However, Mr. Pi does not believe knowing the technology is important, rather it can be figured out as you go: "...Like that's real life, right…you're handed a project and you gotta figure it out. There's no expert in the room to tell you how to do your job" (second interview). This is something that he believes both students and teachers can embrace in the classroom.

In addition to using technology to make their learning come alive, Mr. Pi also uses technology to foster a sense of community in his classroom and to provide a way for all of his students to feel comfortable communicating. Discord, a communication application, is one of the mediums that students can choose to participate in (both during and outside of class time). He has servers for each of his classes that students are invited to join. Then within each server are different channels (or categories). For example, when incorporating the district's equity statements into class time for evaluation and discussion (see Figure 23), he also provides an opportunity for them to continue the conversation on the dedicated Discord server for the class and within the appropriate channel. As students continue working on their projects, there is also chatter happening in Discord that Mr. Pi also engages in.

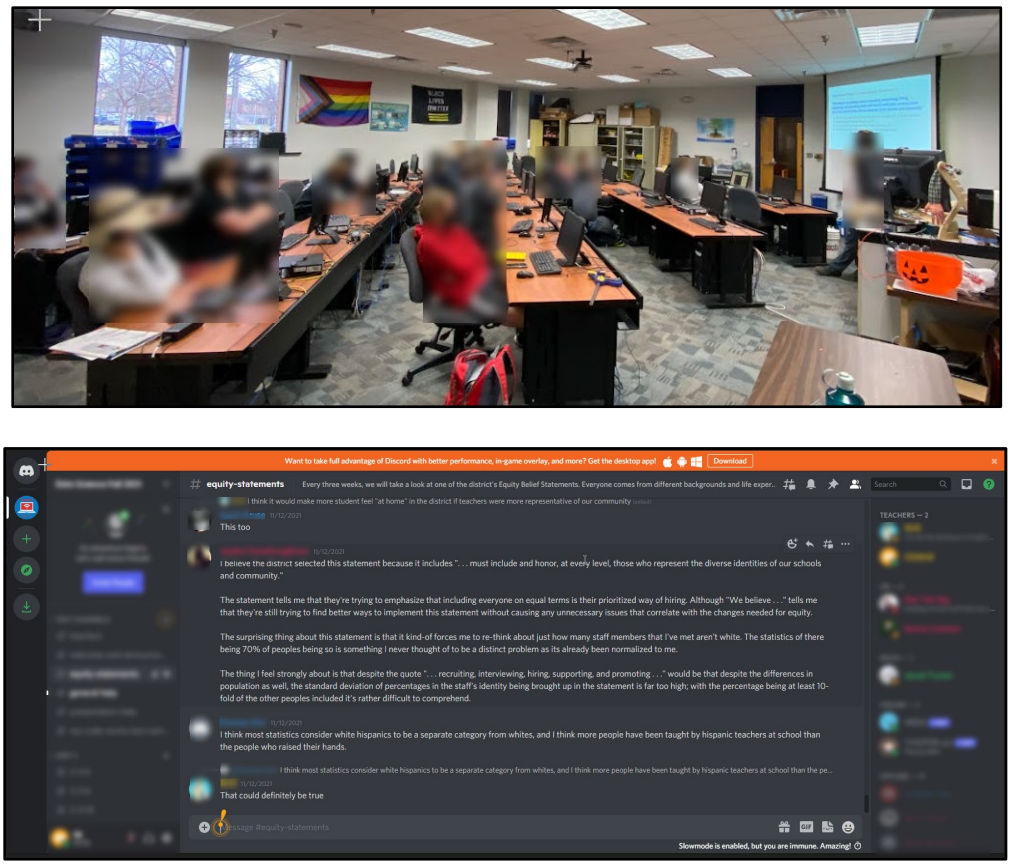


Figure 23. Students discuss one of the district's equity statements. Mr. Pi shares one of the district's equity statements while students discuss it out loud (top image) and virtually via Discord, a communication application (bottom image).

Mr. Pi's approach to teaching with technology is to support rather than lead; he finds that having students use the technology is more valuable than just teaching them about it. To support students in their development of CS and CT he provides hands-on activities, embeds projects that give students ownership, and wants them to use technology so that they can both get immediate feedback and visualize what they are learning. Collaboration, unplugged activities, structured challenges, and time to share/showcase work are some of the strategies he uses when integrating technology into his lessons and classroom.

Summary

While Mr. Pi knows that computational thinking is often touted as a magical ability, he does believe that it is inherent in computer science and can be taught even more explicitly through clear unplugged activities (e.g., writing an algorithm on how to tie shoes) and through encouraging more reflection from students on how they solved computational problems or debugged while programming. As a high school computer science teacher, he tries to find ways to excite curiosity and expose students to as many technologies as possible in the introductory courses. Yet it is in the more advanced CS classes that he finds it more difficult to find room for integration, but it is still something he works to improve every year such as working to incorporate more social justice discussions and applications to his data science class.

Not only does Mr. Pi want students to experience the same fun and enjoyment that he finds in CS, but he now feels an obligation to provide opportunities for students to not just have access to CS, but to learn computer science in a way that empowers students. Since Mr. Pi believes computer science is an incredibly powerful tool that every student should be familiar with, he advocates for it to be integrated into the elementary classroom. He believes that making connections between CS and other subjects are where transference can happen for students, and what better way to do that than integration. Yet, he is also familiar with the challenges that come with this such as no state standards, lack of CS licensure, and the fears that teachers have around CS and technology. In addition, he wants students to have fun opportunities to get "hooked on" CS, so he is weary of only AP CS courses being offered at schools. Despite his realistic views that CS is likely being

taught in most high schools with a prescriptive, esoteric approach, he is hopeful about the ever-evolving CS lessons and resources becoming available.

The role of community is a huge part of Mr. Pi's TPACK-CT - both for his students and himself. He prioritizes ensuring that all students feel they belong and creating community via "forced interactions" like assigned group work and utilizing digital spaces as another way for students to communicate and belong. Not only does Mr. Pi want his students to embrace more communal goals rather than individual ones, but his value of community is also visible in how he connects with others. Even with his strong background and expertise in CS, Mr. Pi still looks to his communities to grow his own practices. He is a part of multiple networks and supports such as professional CS teacher organizations, professional development connections he has made as a participant or leader, online resources and conversations, and gaining insights from his instructors, peers, and research as he advances his own scholarship. All of this continues to shape what drives him to teach CS and what he values most in the classroom.

Technology is one way that Mr. Pi has found to create community within his classroom. Not only does he use the space it can provide for students to connect, but he also encourages students to figure it out by searching and/or working together. He sees technology as something that can support learning and bring CS to life for students. By using project-based learning strategies in combination with technology he is able to encourage students to learn unfamiliar tech, work together, go deeper in their learning, and have tangible ways to share their learning with those around them. While Mr. Pi's

courses use many different technologies, it did not feel like that was the center of his classes or our conversations.

Mr. Pi uses technology as a support within CS to empower students to make choices, collaborate, embrace ambiguity and figure things out, and as a tool for students to demonstrate/showcase their learning. His projects are not about getting it right or even having a working end-product, but rather the focus is on what the student learned. This is something he also believes teachers new to CS or being voluntold to teach CS can embrace: it is okay to not be the expert or not to even be familiar with the technology - just do your best in the moment.

CHAPTER 7: CROSS-CASE FINDINGS

After conducting a within-case analysis for each participant and sharing the findings in the previous chapters, a cross-case analysis was completed to "build abstractions across cases…[and] a unified description across cases" (Merriam & Tisdell, 2016, p. 234). This cross-case analysis addresses this dissertation's second research question, *How do teachers' TPACK-CT compare and contrast across cases?* The thematic findings are presented as five overarching themes which are supported by detailed sub-findings to further expand upon how teachers' TPACK-CT compares across participants. The following themes guide the remaining sections of this chapter: 1) Teachers need supports to face the complexities of computing integration; 2) participation in positive professional learning experiences is instrumental for developing a teacher's TPACK-CT; 3) teachers conceptualize computational thinking as a valuable and transferable skill that can be taught in K-12 curricula; 4) using student-centered strategies and emphasizing connections are two guiding pedagogical strategies for computing integration; and 5) technology is a support for - not the heart of - integrating computing.

Teachers Need Supports to Face the Complexities of Computing Integration

Each of the three participants had varying contexts that account for some of the differences in their approaches to enacting computing integration. There is also no intended curriculum for computational thinking (CT) and/or computer science (CS) in Minnesota (i.e., set of standards). For example, as a STEM teacher who works with kindergarteners through fifth graders, Mrs. Bee has a lot of flexibility in her curriculum

and does not need to supply grades, but she must plan for a wide range of students that she does not see every day. She also does not have fellow elementary education STEM teachers to align or collaborate with. In comparison, Mr. Ozo has the same group of fourth graders that he sees every day, but he must also ensure his curriculum aligns with the academic standards across content areas for fourth graders so he uses CT/CS as a vehicle and lens for approaching his lessons. He also has other fourth grade teachers in his building to plan with and collaborate. Lastly, while Mr. Pi also does not have standards for his subject matter (computer science), he does have a more focused/specialized curriculum as he teaches semester-long advanced computer science courses for high school students that he sees for over an hour every day. For him, the focus is more on CS than CT since he believes CT is more implicit in his classroom. While there are other computer teachers in the district, they do not engage in much collaboration. Despite their differences in contexts and enacted curriculum, they all shared a common need for supports to face the complexities of computing integration. The absence of computing standards, the need for supportive administration, the importance of community and professional development, and the complexities of computing integration made the desire for supports more visible across participants.

The Absence of Standards

Elementary and secondary educators base and design their curriculums to align with state academic standards related to their content and grade level. Since there are no computer science or computational standards in Minnesota (Code.org et al., 2022), all three participants remarked upon how this impacted what and how they teach. When

planning and designing their curricula and planning lessons, all three teachers rely on what they believe connects to available academic standards, content they find important and have found resources on, and/or content they have experienced during a formal or informal learning experience. A few of the most commonly mentioned resources or websites that the participants mentioned looking to for help with CS/CT activities included Code.org, Scratch, robot-/tool- specific websites, and Google CS First. Some of the teachers also lean on subjects that do have standards and use those as a foundation. For example, Mrs. Bee leans on the math and science standards when designing projects and units for her students, and then includes CT/CS elements in those projects. In comparison, Mr. Ozo tries to view the academic standards through a CT lens and use CT as a way to deliver the content - often pointing out connections from the subject/activity to CT and/or incorporating CS elements, allowing him to weave it in that way. Mr. Pi leans on his own experiences, knowledge of advanced courses, and experience in the industry when designing his courses so he appreciates the flexibility of not having standards. However, he also notes how not having standards means that each district/school/teacher is doing their own thing, which means even if CS is offered, not all students may be receiving equitable learning experiences (Coenraad et al., 2022; Sadik, 2017). Ultimately, having to find resources and approaching integration without standards can make planning and integration more difficult as teachers must search for resources (often on a subject they may not have a background in) and it may cause them to feel more isolated as they struggle to collaborate with other teachers because there is no shared foundation (Sadik, 2017; Yadav et al., 2022).

Not only can a lack of academic standards make planning and integration more difficult, but it can also lead teachers to judge their own enacted curriculum or fear that they may get questioned for bringing in something that is not technically required. For Mr. Ozo, a lack of standards meant that there is no benchmark for him as a teacher, and at times has led him to feel like he isn't doing "enough," particularly when he compares himself to middle school teachers. There is no standard to know if you are meeting it, so teachers place their own expectations on themselves and/or compare to what others are doing (often in completely different contexts). Mrs. Bee also expressed a desire for wanting to get ideas to improve upon her integration of CT and wishing she had folks to collaborate with in her district, but knows that the other teachers in similar roles at the middle and high school are teaching completely different content/have different focuses.

Standards can serve as a support that aids teachers in feeling free to teach and integrate computing. Both Mr. Pi and Mr. Ozo commented on the sense of "permission" that standards provide teachers to integrate computing in their curricula. Without standards, teachers may not only fear they are not doing enough or a good job, but they also worry about the perceptions of their school/district administration and not feel confident in defending their actions if questioned.

The Need for Supportive Administration

Supportive administration provides teachers with a sense of approval to integrate CT/CS into their classrooms (Margot & Kettler, 2019). In a time where teachers are often being challenged and curriculum is being revised based on political agendas (Woo et al., 2023), it makes sense that in a state with no CT/CS standards, teachers may have some

apprehension about bringing it into their classrooms. Support by principals and other school/district leadership was mentioned by all three teachers and the feeling of support seems to connect to their comfort with trying new things and making modifications to their curricula. The teachers all have anecdotes about seeing schools and districts take on initiatives that eventually fade away, never occur, or are replaced by the next new thing; so, they also know that it can feel difficult or scary to buy-in to a new initiative and/or to put the time into altering the prescribed curriculum because it is always evolving. However, these teachers felt enough support and safety to integrate CS/CT into their existing classes, rather than having to wait for a school-wide initiative.

Mr. Ozo knows that to fit everything in that his school asks him to, he occasionally has to pause or deviate from the outlined curriculum. He feels confident in his experience and supported by his school leadership so he does not worry he will be questioned on this or "get in trouble;" but, he knows some of his colleagues cannot imagine skipping over sections of the prescribed curriculum. Since Mr. Ozo also has experience seeing how things can connect for students (e.g., CT integration has improved students' word problem skills in math), he feels comfortable trying things and saying yes - and safe to share those decisions with his principal. Mrs. Bee also feels supported because the superintendent initiated the original creation of her STEM role. She also felt support from her building principal when she moved into her current role. She feels free to try things, pursue learning experiences, and continue to work to integrate computing into all of her projects. Some of this support also leads to feeling like she has autonomy in her role, as well as being supported with resources and donations (e.g., Legos, toilet

paper rolls, dominos, etc.). Mr. Pi mentioned his appreciation for his leadership and that he feels like he has more freedom because his role is not housed in a high school within the district. He also brought up that because he teaches something that not everyone is familiar with, he feels like there is less chance of school leadership coming into his classroom and questioning his curriculum/activities. Not only did all three teachers express feeling supported by their school leadership, but their administration also showed support when they approved their participation in this study.

The Importance of Community and Professional Development

All three teachers mentioned finding support and resources outside of their schools in learning communities and/or in their professional development experiences. During interviews with all three participants, it was common for them to bring up an anecdote of something they read on social media, share a conversation they had during a conference or professional organization meeting, and/or recall something that happened during a professional development experience. Connecting to other teachers, whether online or in-person, expanded these teachers' communities and alleviated feelings of being siloed (Sadik, 2017). One of the benefits of these expanded communities was exposure to different frameworks and research around integration and computing. Mr. Ozo's networks not only gave him a new lens to view integration through (exist, enhance, and extend) (Waterman et al., 2020), but participating in a conference also helped him acknowledge how much CT integration he was actually doing in his classroom. Mr. Pi's communities have also exposed him to new computing research which has impacted his approach to teaching computing including getting new ideas for his own teaching practice

from his postgraduate studies. In addition to online social networks, Mrs. Bee stays connected with her previous professional development cohort and the incoming cohorts which provide her opportunities to reflect on how much she has built in, to get new ideas, and to provide mentorship to others in her state looking to teach/integrate CT/CS.

All three teachers also commented on the different ways their PD experiences and connections impacted or ignited their motivation for teaching CT/CS, as well as provided them with resources to aid in the process. Not only did these teachers' PD experiences enhance their knowledge of CT/CS, but they were integral to helping teachers address misconceptions (about themselves and about CT/CS), in addition to providing community and curriculum. Mr. Ozo's engagement in PD allowed him to not just see himself as a coder, but to also see the value of providing quality coding opportunities to his students at a young age. Mrs. Bee gained a community as well as more ideas and resources beyond Scratch that she has continued to weave into her classroom. While Mr. Pi had a background in CS, PD also expanded his ideas about equity, research, and shifted his motivation for teaching CS. As CT and CS continue to gain momentum in the education landscape, it seems critical that teachers continue to engage in communities and learning experiences to avoid feeling siloed, to provide ideas/resources, and to help illuminate the progress they are making - one step at a time (Hubbard, 2017; Jocius et al., 2021; Kong et al., 2023; Margot & Kettler, 2019; Zhu & Wang, 2023).

The Complexities of Computing Integration

All three participants articulated that while integration of CT/CS is the goal, it can be difficult to achieve true integration. The value that all the teachers saw in

computational thinking was its transference to other areas, so they felt that to ensure "computational thinking" was not just a buzzword, it would need to truly be integrated for those connections to be made and not be taught as an isolated subject. Similarly, computer science was also discussed as a tool that can be integrated across subjects to illuminate how powerful of a tool it is and its applicability across content.

One of the opportunities that comes with integrating CT and CS is that it means something else does not need to be discarded in order to fit it in. Rather, as Mr. Ozo points out, CT can act as a vehicle to deliver content and make the classroom feel more cross-curricular as connections are made amongst different content areas (Margot & Kettler, 2019). For example, he will weave in games that are about debugging and patterns, and then reference them as students engage in math problems. He might also bring CT into the time spent on English Language Arts (ELA) through having his students practice conversation with sequenced sentences and on a topic that is CS-related. At the end of the day, he will have a relevant, short video clip related to their learning and some of the CT/CS practices and concepts the students engaged in throughout the day. Mrs. Bee finds weaving in CT/CS to her units and culminating projects enables her to make content aligned with existing academic science and math standards, to provide choices for students and options to enhance projects, and to provide fun, engaging computing opportunities for her students. So, for example, while students are learning about pollination through hands-on activities and stories, they are also learning about algorithms with small groups in order to program their Bee-bots to "pollinate" some of the flowers while also synthesizing what they have learned thus far. Mr. Pi also discussed

the ways that CS in particular overlaps with social studies and art and how basic, unplugged activities in math can teach algorithms and sequences. All three teachers also identified inherent areas where both teachers and students may be unaware they are practicing CT (e.g., multi-step directions, solving math problems, debugging programs, etc.).

There are also obstacles in integrating CT/CS that were highlighted during the study. For example, it is important for teachers to be aware of what CT and CS are, but actual experience with integration is also critical. All three teachers pointed out the time constraints that teachers feel to fit everything in and the fear of deviating from the prescribed curricula. Mr. Ozo reflected on how even coaches who come to help teachers in his school adapt and adopt new standards are just incorporating them by writing them on the board, which he asserts is not integration (Beane, 1995). Teachers that do not feel supported by their administration to weave in CS/CT and/or without academic state CS standards may not feel that CT/CS is valued or a priority. Or as Mr. Pi shared, the teachers in his district who teach CS are often math teachers who were "voluntold" to teach it. While they move forward with teaching the AP courses, they are usually not willing to invest more time in CS via collaboration or PD. He admits this could be a loss for students as the teachers may not have had exposure to quality computing experiences (as a teacher and/or as a learner) where the course was designed for students to be engaged and have fun. All three teachers discussed that their schools had not provided professional development opportunities on neither integration nor CT/CS, so unless a teacher personally decides to seek them out, teachers remain unaware of the possibilities.

Ultimately, multiple supports are necessary for teachers to navigate the complexities that computing integration brings.

Participation in Positive Professional Learning Experiences is Instrumental for Developing a Teacher's TPACK-CT

When comparing the participants' TPACK-CT, their positive experiences with professional learning was instrumental in how they approached integrating computing into their contexts. The key role that professional learning experiences played was evident as all three teachers reflected upon their prior computing PD, a continued commitment to professional learning, and their prioritization of working toward equitable computing experiences.

Prior Computing Professional Development

All three participants had diverse backgrounds and experiences: two (Mr. Ozo and Mr. Pi) had educational and industry backgrounds in STEM careers (engineering and computer science) and one (Mrs. Bee) had a long career in teaching elementary education. Yet each teacher also had some type of positive and impactful computing professional learning experience, too. Often when reflecting on their teaching practices and beliefs, the influence of these CT/CS learning experiences was visible - from the confidence they had in their understanding of computing and their ability to teach it to their reasons for why they wanted to teach it. Despite their array of experiences, they each had participated in a PD that they found influential in teaching and integrating CT/CS, and were able to recount how these experiences influenced their beliefs (about teaching, about computing, and at times about themselves).

Teachers teach what they know (El-Hamamsy et al., 2021) and those practices continue to be influenced by many factors (e.g., their district, their administration, life circumstances, etc.) (Hubbard, 2017; Jocius et al., 2021). These participants discussed how their teaching practices continue to be shaped by additional learning experiences. Talking with the teachers made it more evident that they are (sub)consciously reconciling their ingrained beliefs, histories, and experiences with new learning and beliefs. For example, Mr. Ozo reflected on how his strong background in CT was influential in navigating his own path through education and career, but it was not until a PD experience where he first saw himself as a coder that he realized students are capable of coding at a young age. Mr. Pi's continued education made him emphasize community and belonging in his classroom - he came to see relationships as the main goal rather than just replicating teaching/curriculum that he found to be successful for himself as a student. Mrs. Bee went from only using what the previous STEM teacher provided (Scratch) to weaving in more connections and ideas that she discovered through workshops and PDs, and seeing computing as something to be woven throughout her curriculum rather than as an isolated unit. To avoid getting caught in a cycle of teaching how they were taught, teachers need new experiences and exposure to different approaches and content to continue evolving their own approaches to teaching (Alqahtani et al., 2022; Jocius et al., 2021; Ottenbreit-Leftwich & Yadav, 2021).

Continued Commitment to Professional Learning

In addition to reflecting on their experiences and sharing what their current computing teaching and integration looks like, all three also shared a continued

commitment for professional learning and growth as they expressed hopes and desires for the future. Just as they build in productive challenges that they want their students to find beneficial, these teachers also continually persevere through the challenges they face in teaching/integrating CT/CS - none felt that they had mastered it. Mrs. Bee continues to build on her current curriculum by weaving in new resources and hopes to work on helping students articulate explicit connections between CT and their work as well as finding a way to build in more sharing of projects. Mr. Ozo is not only evolving his curriculum every year with new connections, but he is also trying to think about ways to engage and excite his colleagues. Mr. Pi hopes to make more social justice connections explicit in his classroom and make equity talks a part of every course. Their passion for teaching and learning is still aflame as they continue seeking out learning opportunities and thinking about ways to evolve their TPACK-CT.

Prioritization of Working Toward Equitable Computing Experiences

Each teacher's professional learning has also led to prioritizing equity in their classrooms and computing integration/teaching. A part of what seems to inspire these teachers to continue teaching and integrating CS/CT, despite the challenges that have emerged, is seeing how computing has positively shifted the culture of their classrooms. For Mr. Ozo this has looked like creating a classroom where students' self-efficacy is being positively influenced and failure/debugging is normalized; for Mr. Pi it looks like creating a culture in his classroom where communal goals are just as, if not more important, than individual goals; and for Mrs Bee it looks like students are encouraged to try new things and be successful. Mr. Ozo and Mrs. Bee also work toward showing their

students more computing pathway options (both in school and in careers) - something that helps expand what and who students are exposed to. A lot of these shifts are a result of both becoming more aware of how computing and equity are so intertwined as well as implementing more equitable teaching practices to teach/integrate CT/CS (Gretter et al., 2019; Koshy et al., 2022). All were familiar with research on the importance of providing equitable computing experiences to students before middle school if there was hope of them seeing themselves as successful or good at CS (Ketelhut et al., 2020; Tai et al., 2006), and this caused both Mrs. Bee and Mr. Ozo to want to ensure their students felt successful in computing while having engaging, social learning experiences that they found relevant. Mr. Pi also wanted to ensure there were no prerequisites to take an introduction to computing class so that all students could have access to a computing opportunity, and he intentionally organized it in a way that explores a variety of tools so that students would have fun and have opportunities to connect it with their interests. Mr. Pi has also found opportunities in his high school classes to discuss equity and computing with his students - making it even more explicit. Being aware of the interconnectedness of equity and computing, as well as having learning opportunities to inform ones' TPACK-CT is imperative to ensure that CT/CS are taught in equitable ways.

Teachers Conceptualize Computational Thinking as a Valuable and Transferable Skills That Can Be Taught in K-12 Curricula

During the study, it was easy to see that these participants were united in their conceptualization of CT as a transferable, multidisciplinary problem-solving strategy.

They also expressed confidence in their own knowledge and teaching abilities of it, and expressed that they saw value in teaching computing in K-12 elementary education.

The Transferable, Multidisciplinary Nature of CT

Despite a wide array of experiences, backgrounds, and expertise, all three participants both answered similarly on the CT survey regarding how they conceptualize CT (see Figure 24) and each differentiated CT and CS from each other. They all strongly agreed that CT involves logical thinking for problem-solving and that transference is a part of CT. They collectively did not place a high value on the role of computers in CT, although their strength of belief did vary slightly on if CT is understanding how computers work and if it involves using computers to solve problems.

Both in their written responses on the survey and during the study activities, the following concepts were commonly mentioned: algorithms, pattern recognition, decomposition, and abstraction. Throughout the interviews, the teachers also noted that CT is both concepts and soft skills and/or a problem-solving strategy - all teachers agreed that CT was not just creating a PowerPoint or learning basic computer skills/competencies (e.g., logging in, playing a game, etc.). Interestingly, though, Mr. Pi seemed to have the most nuanced conceptualization of computer science as he did not envision that most of his students would consider themselves computer scientists - perhaps because that is his area of expertise and/or he works with an older age group. He shared how some classes were more about the “pure science” of CS and studying the tools themselves (second interview), compared to the elementary educators who often

approached both CT and CS with the lens of constantly making connections and believed that some of their students would consider themselves computer scientists.

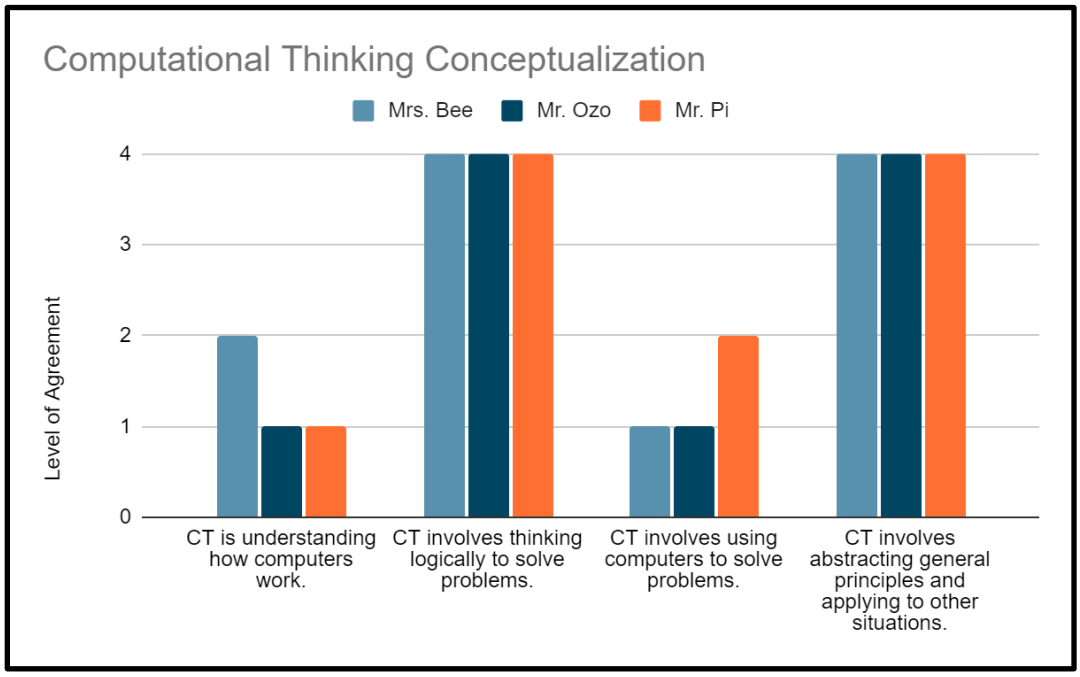


Figure 24. A bar graph showing how the participants conceptualized computational thinking. The three participants responded to statements about how they conceptualize CT on a Likert scale where 1 = strongly disagree, 2 = disagree, 3 = agree, and 4 = strongly agree.

They all also conceptualized CT as a cross-curricular, multidisciplinary skill. While Mrs. Bee and Mr. Ozo, who worked in elementary classrooms, seemed more able to actively integrate it across subjects, Mr. Pi also reflected on both CT and CS being applicable across content areas. All teachers pointed out overlaps of CT with other content (i.e., math, science, ELA, social studies, art) and identified aspects that are similar to other thinking processes like algorithmic and mathematical thinking as well as design thinking. Ensuring that students see connections between CT and CS were important to the three teachers and something they strove to point out in their classrooms.

Confidence in Knowledge and Teaching Ability

On the CT survey all three teachers expressed confidence in their computing knowledge as well as their teaching abilities - including the ability to make connections (see Figures 25 and 26). Mr. Ozo and Mr. Pi were slightly more comfortable than Mrs. Bee in their ability to describe and learn computing concepts as well as in their programming skills. Interestingly, both Mr. Ozo and Mr. Pi have backgrounds outside of teaching and are representative of the demographic who historically have pursued careers in computing. In comparison, Mrs. Bee's background is in elementary education and she is representative of a group that has historically not had high representation in computing. Teachers' histories, beliefs, and experiences influence their own confidence (Ni & Guzdial, 2012).

All three participants also felt positive in their abilities to recognize and appreciate CT concepts across subjects and in daily life, explain computing concepts for effective teaching, and in developing/teaching effective computing lessons. While both Mrs. Bee and Mr. Pi were positive about integrating computing into their current curriculum, Mr. Ozo was not as positive when it came to integration. It can be surmised that the differences in each teachers' background and context could have influenced their responses to these questions. For example, Mr. Pi has a degree in computer science and teaches computing classes so it makes sense that he would strongly agree that computing fits into his content and that he would feel very confident in the subject matter despite that he is the participant with the least amount of teaching experience. Both Mrs. Bee and Mr. Ozo teach in other areas so teaching computing means integrating it for them so they

have experienced challenges in fully integrating it which could have impacted their level of agreement. Also, in comparison to Mr Ozo, Mrs. Bee has more freedom in her role in the sense that she is not trying to cover all academic content areas, just STEM. Mr. Ozo, on the other hand, has an engineering background so making connections to daily life may feel more natural for him as he can view the subject matter through an industry lens. Despite the slight variation in their responses, all three expressed confidence in their abilities to learn and teach CT/CS on the surveys and throughout the study.

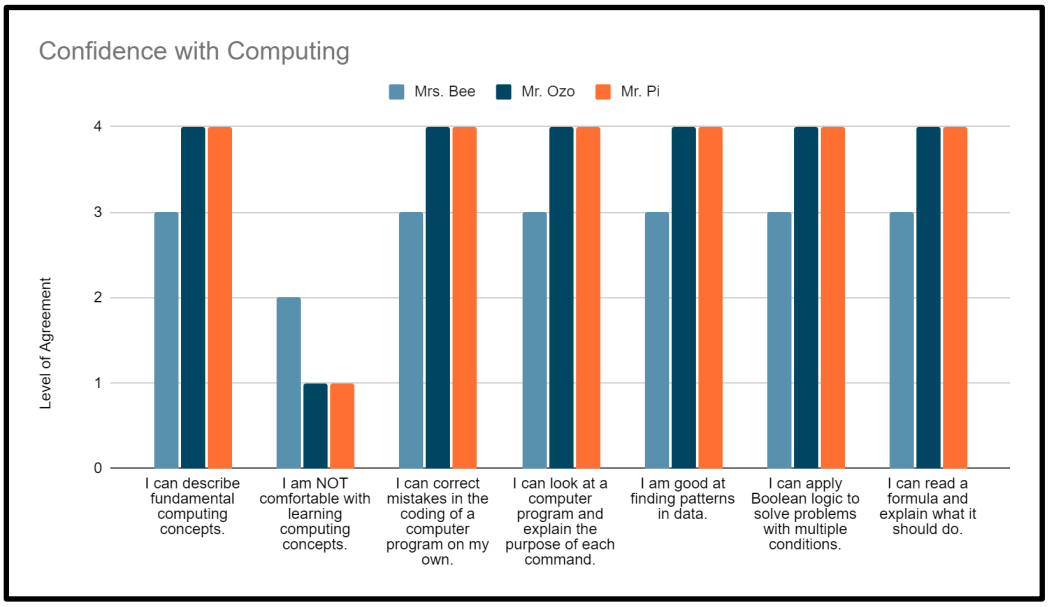


Figure 25. A bar graph illustrating confidence with computing. The three participants responded to statements about how confident they are in their own computing abilities on a Likert scale where 1 = strongly disagree, 2 = disagree, 3 = agree, and 4 = strongly

agree.

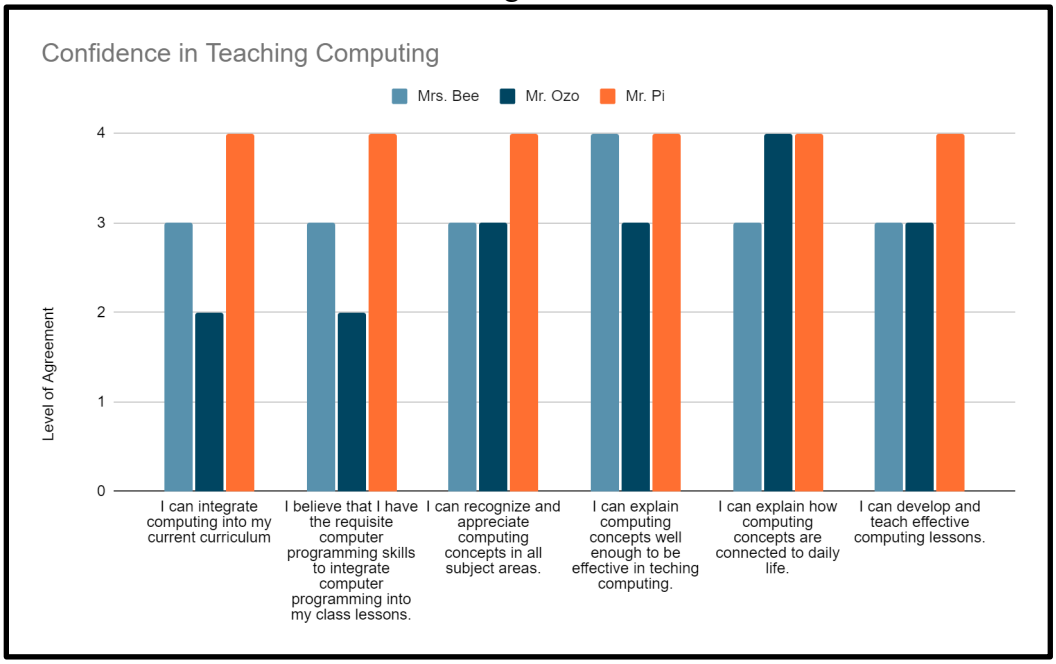


Figure 26. A bar graph illustrating confidence with teaching computing. The three participants responded to statements about how confident they are in their ability to teach computing on a Likert scale where 1 = strongly disagree, 2 = disagree, 3 = agree, and 4 = strongly agree.

Despite their understanding of computational thinking and their confidence in teaching CT, they all mentioned challenges they encounter in weaving it into their curriculum more explicitly. Some of the challenges had to do with a lack of standards and engaging, age-appropriate curriculum available, as well as the difficulty in finding the time to make more explicit connections or to have students share their work.

The Value of Teaching Computing in Elementary Education

Lastly, all participants shared that they feel computing is essential for students to learn - they place a lot of value on computing. Mr. Ozo expressed how he feels it is a critical skill for the students to learn; Mr. Pi articulated that he conceptualizes CS as a powerful tool for historically oppressed students to know in order to empower their

communities; and Mrs. Bee shared how valuable it is to expand possibilities for students and to open up future pathways. In the survey, they all strongly agreed that it should be taught in elementary school and that it can help students become more engaged in school (see Figure 27); however, integrating it into elementary education is challenging as illustrated by Mr. Ozo's feelings on how his classroom lends itself to teaching computing. Similar to their confidence in integrating computing into their current curricula, their beliefs about their current teaching situation lending itself to teaching computing concepts varied based on their role. All three also found computing to be valuable to students' future careers and as an important literacy to have. The value they place on CT is also evident in the pedagogical strategies they employ to teach and integrate CT and their desire for all of their students to have quality computing experiences and exposure to computing pathways.

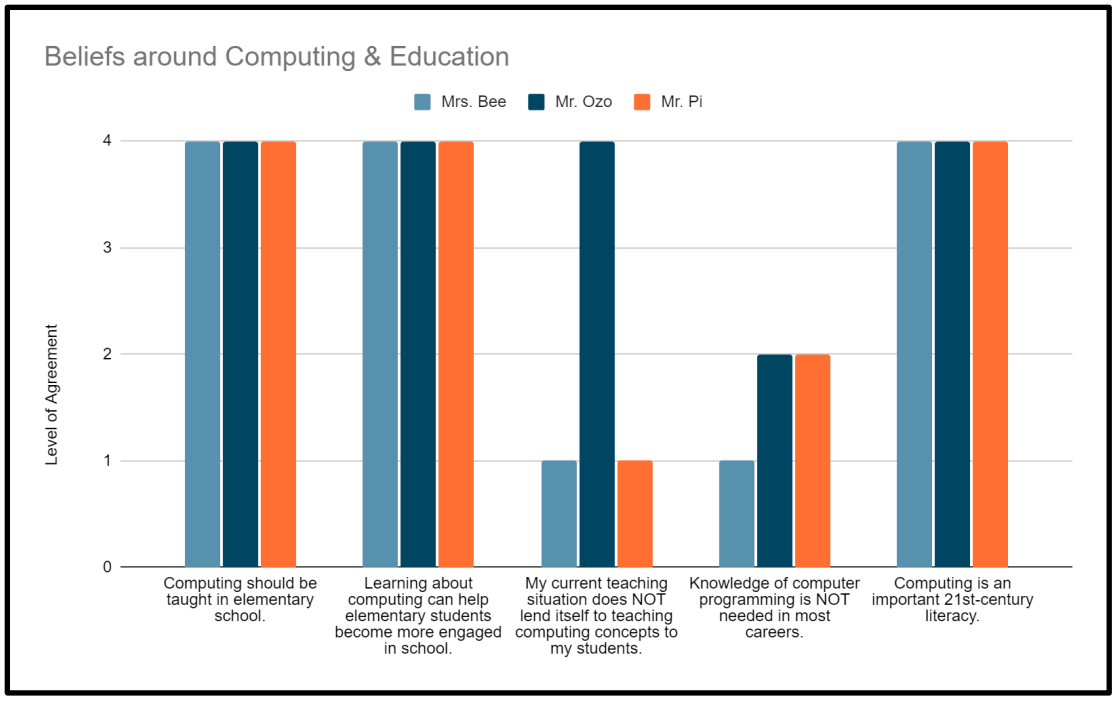


Figure 27. A bar graph illustrating beliefs around computing and education. The three participants responded to statements about their beliefs around computing within education on a Likert scale where 1 = strongly disagree, 2 = disagree, 3 = agree, and 4 = strongly agree.

Using Student-Centered Strategies and Emphasizing Connections are Two Guiding Pedagogical Strategies for Computing Integration

Student-centered strategies and placing emphasis on connections best encapsulate the many shared pedagogical strategies used by Mrs. Bee, Mr. Ozo, and Mr. Pi.

Use of Student-Centered Strategies

All three teachers mentioned wanting to get students hooked or interested in computing. So while their strategies may have differed at times, they were all very student-centered in their approaches to teaching and integrating CT. They each used multiple resources to engage and introduce their students to content in hopes of sparking their interest (e.g., music videos, stories, and examples of finished projects). The power

of using stories was also illustrated as all three teachers employed them to help make content relevant and engaging to students. Teachers also liked how CT invited creativity into computing and the classroom. When projects and activities were used, teachers strove to give hands-on opportunities and multiple customization options for students. While challenges and projects were used to demonstrate students' learning, teachers also ensured the activities/challenges were things the students felt were important (Margot & Kettler, 2019). Strategies like chunking up the information, repetition, and articulating their thinking/learning were all encouraged.

Each teacher mentioned intentionally working to provide students enough freedom to explore, play, and tinker while also giving them some structure and scaffolding to set students up to be successful (Northrup et al., 2022). These teachers would often model activities and options for students, but would try to balance it by giving students productive challenges to struggle with so that they could persevere through debugging and also experience success (Margot & Kettler, 2019). In all three of the classrooms, the teachers did not "do" the project or activity for the student when questions arose, but instead would encourage them to use the resources available, including their peers.

All three teachers did not put a lot of emphasis on grades and assessments when it came to CT/CS learning. Instead, they used more hands-on learning and assessment approaches like project-based learning (Mrs. Bee) or portfolios to demonstrate progress/learning (Mr. Pi). Or they looked to see how students' assessments in other content areas were impacted, as well as looking at the long-term impacts of who chose to

pursue computing pathways at the middle school (Mr. Ozo). Approaching computing this way made space for students to pursue challenges and projects that they were interested in without fear of having a perfect product. Ultimately, all employed student-centered strategies that they hoped their students would enjoy and find engaging.

Emphasizing Connections

In their teaching practices, the participants emphasized connections between computing and other academic content (e.g., connecting CT to other subjects and standards) as well as connecting CT to things outside of the classroom, including career pathways. Not only do the teachers demonstrate their deep understanding of how CT and CS are multidisciplinary, but they work to use strategies that also aid students in making connections from what they are learning to why it's important/applicable. They all taught lessons that built on prior learning and all incorporated small activities/steps to eventually achieve a culminating project. Mr. Ozo often would connect one activity to another that they did earlier that day or in a unit as he and the students moved through subjects. Mr. Pi and Mr. Ozo also wanted to ensure that students could connect their projects and activities to what they believed was important - they wanted students to find purpose in activities to keep them engaged.

In addition, teachers emphasized relational connections by encouraging collaboration. Students in these teachers' classes found themselves encouraged to work in groups, share ideas with one another, help each other, and view computing as a collaborative activity. While the teachers sometimes lamented that they were still figuring out the best way to build in time to share work with peers, they balanced it with some

forced interaction strategies (e.g., sharing with a partner, assigned to groups, having to do a class-wide project, etc.) to encourage community and collaboration. Each teacher really wanted their students to feel successful and like they belonged - all were focused on using collaboration as a strategy to create connections/community in their classrooms.

Technology is a Support For - Not the Heart of - Integrating Computing

Mrs. Bee, Mr. Ozo, and Mr. Pi viewed technology as a way to support and bring students' learning to life - technology was a hands-on way for students to see their learning in action. Each participant used technology as a vehicle to make CT more visible for students, such as seeing how algorithms and sequences impacted the way the program or robot performed, and as a way for students to feel the immediate success when something was debugged (Jocius et al., 2021). Yet all worked to make sure that the technology was not the focus of the lessons, but rather the tool to experience and demonstrate learning (e.g., Mrs. Bee's students used BeeBots to demonstrate their understanding of pollination and Mr. Pi's students were creating a connected, classwide Rube Goldberg machine in which each student had to incorporate and program an Arduino). Participants made sure that their students used technology to practice coding skills and programming as well as to create things rather than to just consume technology (e.g., Mr. Ozo's students used a Scratch fractal program after demonstrating understanding of how it is coded/works).

Another way that technology supported teachers in teaching CT/CS was by expanding the boundaries of their classroom. Teachers were able to expose students to ideas, people, and opportunities via technology. Students no longer had to personally

know someone outside of the classroom to be exposed to possible careers because the internet provided a way for the teachers to bring in different experts and exposure to different careers (i.e., Mr. Ozo did this often via videos and Skype a Scientist; Mrs. Bee used videos; Mr. Pi used Discord).

The teachers' view of technology as a support was also evidenced in the way they used structured play to guide technology introductions and the frequency of unplugged activities utilized.

Structured Play to Guide Technology Introductions

Once students learned the basic skills of using technology, the teachers all strove to provide their students with freedom to play with the technology and experiment with it - particularly when first introduced to a new tool. After giving students freedom to play, these teachers would usually provide some structure and guidelines so that students would have a purpose as they continued to tinker. In addition, students were encouraged to search out answers, ask their peers, and share their knowledge with their class community as the teachers often did not present themselves as the experts in the room when it came to technology (Northrup et al., 2022). Rather than promoting technology as the "end-all" goal of computing, teachers leveraged a balance of structure and fun when preparing their students to engage with technology.

Frequency of Unplugged Activities

Rather than relying solely on technology, the three teachers used many unplugged activities when teaching and integrating CT/CS. Games, partner activities, drawing, physical objects like blocks/Legos, writing, and storyboarding were some of the

unplugged ways that teachers helped their students practice/learn CT and CS skills prior to bringing in technology. These unplugged activities were then connected to a project or activity that brought in more plugged technology tools. Being able to use unplugged activities also helped teachers overcome the obstacles that can arise when using technology such as district technology-use policies, a lack of resources, and time constraints. None of the teachers, including Mr. Pi who taught computer science, seemed to have a device-centric approach to teaching and integrating CT/CS - rather technology served as a support to bring the learning to life for students.

Summary

While these three teachers came from very different contexts, they also shared many commonalities in their enacted curriculum and in the themes that encapsulate their TPACK-CT similarities. Some of the common supports that they had, which appeared to be part of what empowered them to work to integrate computing into their classrooms, included knowledge of some type of intended curriculum (whether that was standards from other disciplines, CSTA standards, PD curricula, etc.), experiences feeling supported in their roles, having community and positive PD experiences, and feeling supported in navigating the complexities of computing integration.

Throughout the study it was clear that each of the teachers' backgrounds impacted their practices and beliefs - and that their positive professional learning experiences had helped them shift their teaching practices to integrate CT/CS. For all of them, equity played a role in this shift and reprioritization of computing in their classrooms, as did

seeing the success(es) their students experienced and their own unique character traits and passion for learning and growing as teachers.

They each shared similar conceptualizations of CT, particularly that it was a transferable and cross-curricular problem-solving strategy. Part of what shaped how they saw CT was also their own confidence in their understanding and ability to teach it to their students - which all of them felt they could do well. They also each believed in the value of integrating CT/CS. In order to do this they all used student-centered pedagogical strategies, including balancing play and structure as they introduced challenges, technology, and projects related to computing to their students. They also emphasized the connections that computing brings - to other subjects, to the world around them, and to careers - as well as the pedagogical decision to ensure computing is collaborative so that students create relational connections. They each used technology to support student learning of CS/CT and emphasized hooking students' interest through structured play. None of the teachers relied solely on technology as the heart of computing since unplugged activities were weaved in and even emphasized at times as more useful in the goal of the learning activity.

Mrs. Bee, Mr. Ozo, and Mr. Pi may have come from different backgrounds and teach different content and grade levels, but they are examples of the teachers in Minnesota working to integrate CT/CS. Their individual portraits as well as the commonalities across them provide us with even more insight into the lived realities of Minnesota teachers working to integrate computing into their curricula.

CHAPTER 8: DISCUSSION AND SUMMARY

Integrating computing into K-12 curricula has the potential to provide learners with early exposure to the possibilities of creation, engagement, and contributions within a technology-driven society (El-Hamamsy et al., 2021; Vivian & Falkner, 2019). In order to help students make lasting cross-curricular and authentic connections through computer science and computational thinking, teachers need training and support to understand, teach, and integrate them within their own classrooms (Moudgalya et al., 2021; Sadik, 2017). While prior research has studied the design and immediate impact of CT-focused professional learning experiences for K-12 teachers (i.e., Bower et al., 2017; Kong et al., 2023; Mouza et al., 2017; Yadav et al., 2014; Zha et al., 2020), more research is needed to understand teachers' translation of these experiences into their own classrooms (Barr & Stephenson, 2011; Bower et al., 2017; Falkner et al., 2019; Guo, 2022; Mannila et al., 2014; Yadav et al., 2014). This chapter will share a summary of my research as I sought to address this gap in literature, discuss my findings and their implications, share recommendations for future research, and address this study's limitations.

Summary of Dissertation Research

Scholars agree that having the knowledge and skills necessary to understand, critique, use, troubleshoot, and create with the technologies prevalent in our society is critical (Czerkawski, 2015). Computational thinking and computer science education, a critical digital literacy, have emerged and been advocated for as one route to prepare K-12 students to develop these necessary skills (Guzdial, 2019; Scharber et al., 2021; Vee,

2018). Integrating computing into K-12 education rather than only offering high school standalone CS classes is one way educators and researchers have tried to promote access for all students and to try to prevent perpetuating existing inequities in computing (Ketelhut et al., 2020; Peterson et al., 2019; Tai et al., 2006; Yadav et al., 2019). As a result of this call and the rise of K-12 CS education, intended curricula (e.g., standards, statewide initiatives, etc.) and professional learning experiences for pre- and in-service teachers have been designed, implemented, and studied in hopes of preparing educators to integrate and teach computing (Codding et al., 2021; El-Hamamsy et al., 2021; Zhou et al., 2020).

In order to better capture and understand the enacted curriculum of how K-12 educators in Minnesota integrated computing into their content areas, my comparative multicase study utilized the technological pedagogical content knowledge (TPACK) framework (Koehler et al., 2013) with the incorporation of computational thinking (TPACK-CT) (Angeli et al., 2016; Mouza et al., 2017) and answered the following questions:

- What does teachers' TPACK-CT look like when enacted in the classroom? *[technological pedagogical and content knowledge related to CT, TPACK-CT]*
    - How do teachers conceptualize computational thinking knowledge in their classroom? *[content knowledge, CK-CT]*
    - What pedagogies are being enacted in the classroom when teaching CT/CS? *[pedagogical knowledge, PK-CT]*

    - What technologies, if any, do in-service teachers utilize for teaching CT/CS? *[technological knowledge, TK-CT]*
- How do teachers' TPACK-CT compare and contrast across cases?

Case portraits for the three participants were constructed after collecting data related to their experiences via a survey, observations, interviews, reflections, and curriculum artifacts. These individual case portraits provided rich descriptions to aid in deeper understanding of the teachers' enacted curriculum and were used to answer the first research question and its sub questions regarding their TPACK-CT, specifically looking at their conceptualization of CT, their pedagogical approach to teaching CT/CS, and their use of technologies to teach computing. To answer the second research question, a cross-case analysis was done to find commonalities and differences across their TPACK-CT. Five themes resulted from this analysis and were expanded upon to identify how these themes connected to teachers' TPACK-CT and its constructs. The following section shares how these findings support current literature and discuss potential implications for practice.

Discussion and Implications of the Findings

The individual cases and the cross-case findings support the literature and scholarship that have already identified some of the challenges of and the supports needed for computing integration (see Table 7). These challenges and supports in the table are organized by the cross-case thematic findings to illustrate where they emerged during analysis. While many of these challenges/supports stood out during the construction of individual case portraits and layers of cross-case analysis, they may not

have been as apparent in Chapter 7's discussion of the overarching themes. This table summarizes some of the broader connections between this study's findings to existing research (e.g., the challenge of infrastructure misalignment was visible when teachers discussed needing to feel support from their administration to fit in computing integration or when sharing the importance of collaboration). Challenges and supports that corresponded are listed across from each other when possible, but it is also more nuanced and interconnected than the table portrays because supports could alleviate multiple challenges or challenges could have multiple supports.

Table 7

A Summary of the Challenges of and Helpful Supports for Computing Integration in K-12 Education that Emerged During Cross-Case Theme Construction*

Theme	Challenges	Supports
Teachers need supports to face the complexities of computing integration.	• **Infrastructure misalignment** (e.g., school technology policies, inflexible curriculum) (El-Hamamsy et al., 2021; Jocius et al., 2021; Ozturk et al., 2018) • **Lack of collaboration** (Sadik, 2017)	• **Feeling supported** in efforts to integrate computing (Hubbard, 2017; Sadik, 2017) • **Aligned school infrastructure** (Jocius et al., 2021) • **Learning communities** (El-Hamamsy et al., 2021)
Participation in positive professional learning experiences is instrumental for developing a teacher's TPACK-CT.	• **Feeling inadequate/lack of confidence** (Hubbard, 2017; Margot & Kettler, 2019; Zhou et al., 2020) • **Prior experiences** resulting in pre-conceived notions of identity/ability (Hubbard, 2017; Ni & Guzdial, 2012)	• **Finding motivation/the hook** to do the work (El-Hamamsy et al., 2021; Fluck et al., 2016; Vogel et al., 2017) • **Lasting & continuing professional development** (Hubbard, 2017; Ketelhut et al., 2020; Sadik, 2017; Zhou

		et al., 2020) ● **Opportunities to see examples and practice** (Alqahtani et al., 2022; Zhou et al., 2020)
Teachers conceptualize computational thinking as a valuable and transferable skill that can be taught in K-12 curricula.	● **Time** (Hubbard, 2017) ● **Lack of innovative curriculum** (El-Hamamsy et al., 2021; Guo, 2022)	● **Curriculum resources** & free lesson plans/software (El-Hamamsy et al., 2021)
Using student-centered strategies and emphasizing connections are two guiding pedagogical strategies for integrating computing.	● **Different conceptualizations of & levels of integration** (Coenraad et al., 2022)	● **Training on/experience with unplugged activities/pedagogies** (Huang & Looi, 2020)
Technology is a support for - not the heart of - integrating computing.	● **Beliefs/skills with technology** (Veletsianos et al., 2016; Vivian & Falkner, 2019)	● **Exposure/support with technology** (Alqahtani et al., 2022; Vivian & Falkner, 2019)

*These findings are interconnected as often the supports are helping to alleviate the challenges.

In addition to seeing how many of the challenges and supports mentioned by the three teachers align with current scholarship, the findings of this study provide further insight into what teachers have found beneficial and what has stayed with them after engaging in PD. The findings also illustrate the challenges in computing integration that

remain. Four unique contributions of this dissertation (*PD as a bridge for buy-in, integration, and permission; the importance of conceptualizing CT for their own classroom*; *(missing) connections in student-centered, collaborative, and balanced pedagogical strategies*; and *the (un)plugged fun and supportive role of technology*) extend current K-12 computing education scholarship as they provide implications and insight for practice, which can lead to advancing K-12 computing education.

Professional Development as a Bridge for Buy-in, Permission, & Integration (TPACK-CT)

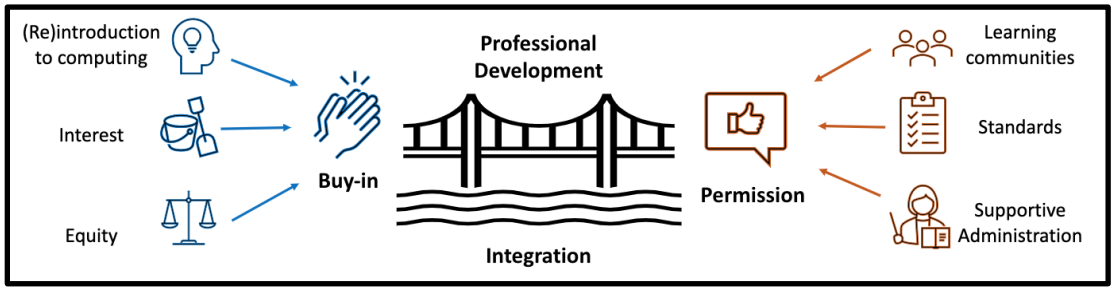


Figure 28. A visual representation of PD as a bridge. It can mediate two contextual factors that influence teachers' TPACK-CT (permission and buy-in) while also bridging the gap that is integration.

Since trying to illustrate a teacher's TPACK-CT also means accounting for contextual factors (Kurt, 2018; Mouza et al., 2017), and because content, pedagogy, and technology were discussed separately in the case portraits, I start by sharing three major components that emerged as a part of the context that influences a teacher's TPACK-CT: *buy-in, permission, and conceptualization of integration and its practices*. Professional development appears to be a way to mediate these two contextual factors that influence teachers' TPACK-CT (permission and buy-in) while also bridging the gap that is integration.

Buy-in encompasses how teachers are (re)introduced to computing, their continued interest, and how they situate equity within computing education. Understanding the possibilities of computing integration appeared to be an important contextual layer to these participants' TPACK-CT. A common thread for these three teachers was that they knew about computing and something had sparked their interest in the first place. In order for teachers to teach something that they may not have experienced themselves or have misconceptions about, they need to become aware of it through exposure if they are going to incorporate it into their classrooms (El-Hamamsy et al., 2021; Hubbard, 2017). While all three participants had different backgrounds and moments that this occurred, they all shared that it did occur. (Professional) learning experiences were one of the most influential experiences that came up for all three as it introduced them to (ways of) computing in the classroom in a way that hooked them and kept them interested, provided them training and confidence in their abilities to teach it, and influenced their motivation for why they wanted to integrate CT into their classrooms. PD appears to be a way to be able to get teachers to buy-in, build their confidence, and sustain their interest by helping them identify why teaching computing is important to them (Codding et al., 2021; El-Hamamsy et al., 2021; Zhou et al., 2020). With equity at the core of these three participants' reasons for wanting to integrate computing, they seemed willing to persist each year by trying new things and pursuing more learning.

Permission encapsulates the role of standards, administration, and learning communities as validation/approval are often provided/felt by the presence of standards,

supportive administration, and (reinforced) amongst peers in learning communities/via PD. The U.S. public education system is built around standards (which are assessed via high-stakes testing), so K-12 teachers are taught to align and prioritize content with academic standards (Margot & Kettler, 2019; Ozturk et al., 2018). These three teachers were no different in that they all referenced standards and the importance of them in their planning to teach computing. All three also mentioned the "fear" that many teachers have to deviate from the prescribed curricula their schools have (Guo, 2022) - they do not know what can be cut to fit in computing, worry about being questioned, may not feel confident in their computing abilities to find ways to weave it in, and/or may feel overconfident in their understanding of computing terms but miss the nuanced complexity of CT/CS concepts and practices.

Something that was surprising and that was not discussed in the prior research I explored is also how the lack of intended curriculum contributed to teachers' feelings of inadequacy or concerns about "doing enough." This may be due to Minnesota's unique context where standards are being integrated into existing ones, as opposed to states implementing standalone CS standards where much of the research has been located. Not having clear benchmarks means that teachers may either compare themselves to other teachers (e.g., online, in professional organizations, at conferences, etc.), their personal set of expectations, and/or both. If teachers are already worrying if what they are doing is okay, not being able to feel that they are doing it well or meeting expectations is difficult. This uncertainty was not only visible in my conversations with the participants, but also

could be seen during the recruitment process where many teachers shared with me "I'm not sure if this counts" when asked to share an example of how they integrate CT.

Something all three participants referenced during our time together was their gratefulness for their administration. They all cited how they felt supported currently or in the past by principals and/or the freedom they find in their roles. They also shared examples of colleagues who had been micromanaged in the past, had negative experiences or didn't feel supported, and the fear that resulted when contemplating doing anything other than the school's prescribed curriculum. Having supportive administration (currently and/or in the past) and being engaged in learning communities (whether virtually, in person, or both) was integral for the three participants to first begin and then continue/encourage their integration of computing (Loepp, 1999; Margot & Kettler, 2019).

Unfortunately, the lack of clear unifying intended curriculum in Minnesota means that districts, schools, and individual teachers are left to interpret and enact what they believe the woven in CS standards means while simultaneously worrying if they are doing it right and "how much" is enough. In addition to feeling supported by administration and aligning to and referencing other academic content standards, professional development appeared to mediate the lack of permission from the CS/CT standard gap. PD provided the teacher participants with both opportunities for buy-in/interest and some type of intended curriculum - whether that was specific, prescribed curriculum like PLTW, an introduction to CSTA's CS standards, ideas for weaving in computing, places to find curriculum online, or shared curricula amongst the members

that gave the teachers something to start with/work toward. Having access to peers also created networks of support for teachers to connect with and to feel less alone in their work. The participants attributed their PD experiences for their own and their peers' increased understanding, pedagogical strategies, specific curriculum ideas, and/or feeling supported/connected. In addition, PD helped them cultivate their own reasons for why they want to teach computing, which may be motivating amidst spaces where it does not feel prioritized (Northrup et al., 2022).

Integration is a means to bring in computing no matter the curriculum/classroom (or even as a way around permission), but it also comes with its own challenge. Comfort with and understanding of integration also seemed like a contextual consideration that is sometimes overlooked, particularly the focus on the actual art or practice of integration (Margot & Kettler, 2019). The three participants' experienced backgrounds and deep knowledge of content standards were visible as they discussed and enacted integration, as well as how they pointed out the possibilities that were a result of integration due to the flexibility and the cross-disciplinary nature/power of computing (Huang & Looi, 2020; Margot & Kettler, 2019; Merino-Armero et al., 2022). Yet it was also clear from anecdotes shared by the participants that integration itself seemed to be a knowledge gap for many teachers, as well as their own articulation of building on the ways they integrated over time, and their personal desires for more specific curriculum and ideas for integrating CS/CT. Perhaps bringing in ideas and examples for how CS teachers at all levels can also integrate in other content areas could be helpful so that teachers who identify as "only the CS teacher," have examples and exposure to other content area

connections, as well as practice integrating (Veletsianos et al., 2016). Being able to see the possibilities of integration is not enough - teachers need to be able to make those connections and enact them in order for their students to experience the cross-curricular connections of computing.

Implications for Practice…

For MN Policymakers. In an education system where standards are the norm, teachers look to them to structure what to prioritize and teach, as well as guidelines to use when measuring what they are (expected to) covering (Guo, 2022; Guo & Ottenbreit-Leftwich, 2020). If computing is going to be embedded in K-12 Minnesota schools with the current approach of weaving it in existing standards, there needs to be a way to ensure *all* teachers are aligned on what that means and feel that it is a priority - they need permission to try. If it remains left up to the teacher or the school to seek out "permission" via supportive administration, attending PDs, and finding learning communities on their own, then it will continue to impact only the students attending that specific school or person's class. To bring equitable computing to all students in Minnesota, it might look like the state providing more guidance or forms of "permission" that are accessible to all teachers such as statewide PD sessions, example curricula, a repository of computing curriculum resources, and a statewide community network for teachers.

For PD Professionals. Since ensuring all teachers have standards or supportive administration or district permission or learning communities feels slightly improbable, it appears that PD can be one way to be the integrative bridge mediating permission and

buy-in. However, in addition to just getting teachers hooked or making computing interesting to them, PD should also include integration theory if the goal is to integrate computing across the curriculum. In order to be this bridge, the PD design should intentionally address these layers of permission and buy-in by ensuring teachers leave with achievable benchmarks (or intended curriculum in mind), a supportive learning community to lean on and be validated by, knowledge of integration models and practice integrating, an understanding of equity's role in computing, and an articulated reason why teaching computing matters to them.

Importance of Conceptualizing CT…for THEIR Own Classroom (CK-CT)

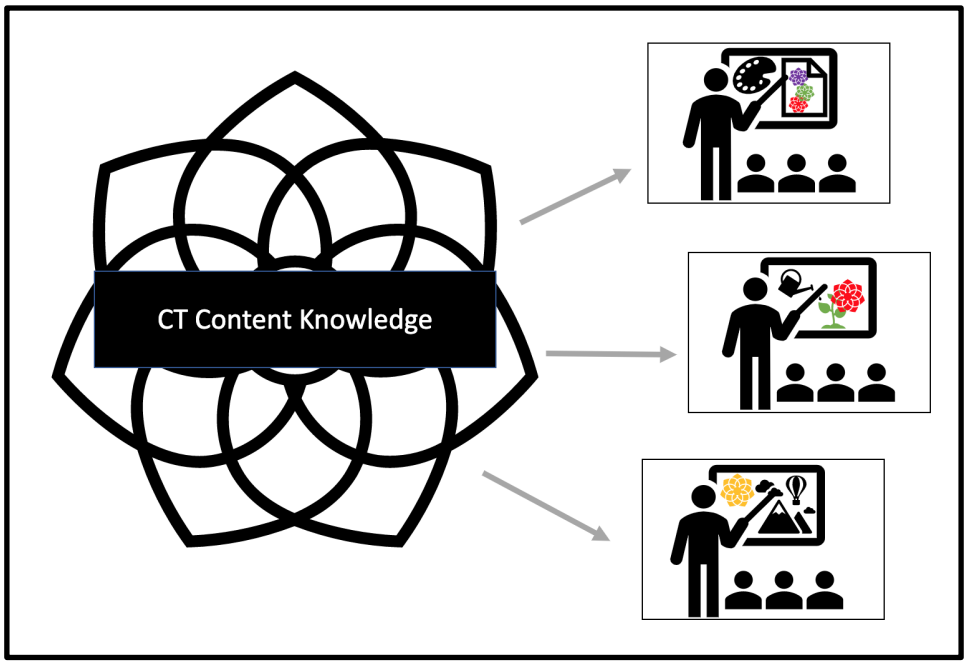


Figure 29. A visualization of CT content knowledge conceptualized differently. Teachers may share a universal conceptualization of CT content knowledge (the larger image on the left), but then their conceptual understanding looks differently when it is translated into their own classrooms (three images on the right represent three different translations).

The three participants of this study were aligned in their CT content knowledge, including how they conceptualized computational thinking - emphasizing that it is a problem-solving strategy consisting of both concepts and practices and that it is transferable and transdisciplinary. They all shared being somewhat confident in their ability to teach it and in their knowledge of computing, and they each found value in integrating computing into K-12 curricula. Yet what differed was how they saw/enacted it fitting into *their* own classroom contexts and the specific ways it can be integrated (which may also be connected to their content knowledge of their subject area expertise and their knowledge about integration). Just as the participants called out over and over, CT is not a magical or mythical term - it does not automatically result in students being able to transfer computing concepts and practices into other subjects (Sadik, 2017). Just as these teachers discussed needing to help their students make connections to how computing relates to other disciplines/the world, teachers themselves need help to make specific connections between their conceptualization and understanding of CT AND their classroom (including content area).

While the computing survey used in this study that was developed based on Yadav et al.'s (2014) Computing Attitudes Survey and Mouza et al.'s (2017) and Leonard et al.'s (2018) adaptations of it provided insight into how these teachers conceptualized CT and asked general questions about how they saw it within their curriculum, I do not believe it captured the nuanced nature of how teachers perceived or made sense of the ways they envision their CT content knowledge and conceptualization fitting into their classroom. It was the observations, interviews, and asking questions that

allowed me to see that while yes, they shared a similar conceptualization/knowledge of CT and high confidence in teaching it, as well as having had exposure to PD experiences, they ultimately conceptualized and enacted it differently within their classrooms.

Implication for Practice…

For PD Professionals. Collaboration, connecting teachers in similar roles and with others who have different ways of viewing how CT fits in their classrooms, and ensuring teachers' can conceptualize and articulate how CT fits *in their classrooms* may aid in ensuring students have more equitable computing experiences across classrooms, schools, districts, and states. As we grow in our understanding of different ways teachers conceptualize how computing fits into their context and curricula, teacher educators and PD professionals can better support them with resources and learning opportunities that are aligned to their specific needs and gaps in where/how they connect computing to their classroom, as well as expand the boundaries/shape of their conceptualization.

(Missing) Connections in Student-Centered, Collaborative, & Balanced Pedagogical Strategies (PK-CT)

Figure 30. A visual of (Missing) Connections. The puzzle pieces illustrate a key theme that emerged within teachers' pedagogical knowledge for integrating computing: the importance of (missing) connections.

All three participants used student-centered pedagogical strategies (e.g., hands-on, project-based, student choice, customization options, etc.) when teaching and integrating computing. They often spoke of wanting to get students hooked or interested so they intentionally chose strategies to align with those goals - such as highlighting the end product, using music videos, and sharing stories. The creativity and collaborative elements of computing (Barefoot Computing, n.d.; Vogel et al., 2017) were often highlighted and brought into the classroom because students were able to bring their interests in and were assigned to work in groups or partners. In the midst of all of the creativity and the choice, these teachers strove to balance it with some structure, scaffolding, and challenge so that students had something to work toward and to ensure students would experience success (Codding et al., 2021; El-Hamamsy et al., 2021). Yet for these teachers, many of these strategies were ones that they used throughout other

content, too. The one strategy that was solely emphasized for and unique to computing was making connections.

The fact that all teachers felt elements of computing were inherent and that some teachers may be teaching it without knowing also means that the missing piece was often making those connections (for the teachers and for the students). Memorizing vocab, an introduction to CT, and a few lessons with Scratch does not mean that the teachers or the students are going to transfer that knowledge across subjects or into their problem-solving skills. To avoid using and framing CT as a mythical term, it is essential that teachers can make the connections to how it's transferable so that they can also help their students make those connections (Coenraad et al., 2022; Vivian & Falkner, 2019). All three teachers discussed how their computing curriculum and areas to integrate grew each year as they reflected, accessed more resources, and had compounding learning experiences. With years of experience, they each shared ways that they helped students make explicit connections, while also noting that this is an area they would like more resources. This also intersects with how teachers conceptualized CT in their classrooms - being able to make connections between CT and content areas, as well as the world, reinforced with knowledge of and practice with integration, are key in order for them to be able to make those connections come alive for their students.

Implications for Practice…

For PD Professionals/Teacher Educators. Training on student-centered strategies around computing is helpful for teachers to begin seeing the collaborative nature of computing and to overcome any initial preconceptions (Jocius et al., 2021;

Sadik, 2017; Veletsianos et al., 2016). Hands-on strategies, building on learning, providing interest hooks and customization options, as well as meaningful challenges are a few of the ways that these teachers found success in initiating and sustaining student engagement (Guo, 2022; Peterson et al., 2019). However, pedagogical strategies along with the knowledge of connections are a key part of computing that needs to be highlighted and supported for teachers. For students to see and apply the transferability of computing, the teachers need to see and enact it in their classrooms, too.

The (Un)Plugged Fun & Supportive Role of Technology (TK-CT)

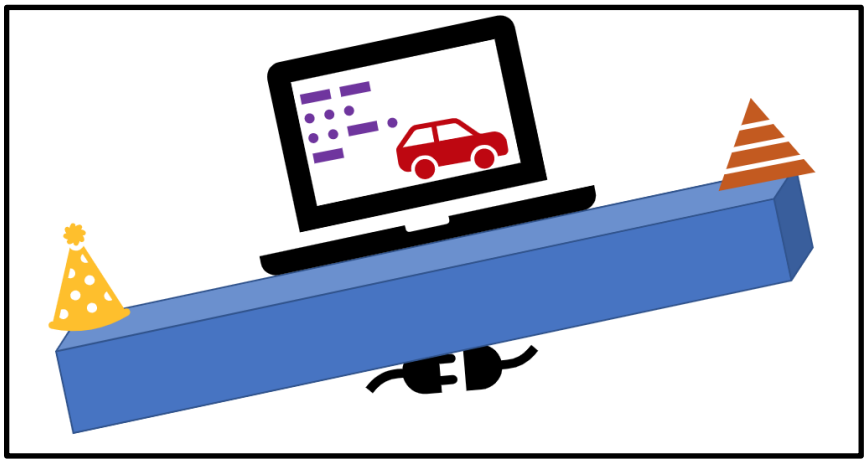


Figure 31. The balance of structure and fun when using (un)plugged technology. A representation of how the participants viewed (un)plugged technology as a vehicle to make computing come alive, but still had to find the art of balancing structure and fun.

All three participants found technology to take on the role of support in their teaching and integration of computing (Vivian & Falkner, 2019) - they used it as a way for students to see their learning in action. Despite the fact that CS and CT may have a connotation with computers/devices, it was not the case in these three classrooms. Instead, these teachers employed technology with the mindset of engaging students in fun and as a vehicle for learning. Sometimes this included using unplugged technology

and/or activities (e.g., blocks, Legos, "being" the computer, drawing, etc.) and it was common to see multiple students gathered around a computer/laptop or leaning across their station to talk to another student - collaborative pedagogy strategies were highlighted when devices came out in these rooms.

All three teachers shared a familiarity with technology and appeared to be seen as technology experts by other colleagues (illustrated when teachers would come to ask them a tech-related question during my visits), but they all also shared a similar mentality of "technology expertise is not required for CT and computing," and believed that they did not need to be the expert in order to use technology (Northrup et al., 2022). They were comfortable letting students tinker, having students help each other or use outside resources (e.g., internet search engines), using already made resources/instructions, pointing to guides if students wanted to enhance or advance their creation, and they practiced those strategies themselves. If teachers are going to use technology as a support and to bring fun into the classroom, they need to also practice and model that mindset for themselves.

Implications for Practice…

For PD Professionals/Teacher Educators. Ensuring teachers have experience with technology that replicates the desired experience for students is helpful in disrupting preconceived beliefs/expectations/identities around technology (Jocius et al., 2021; Margot & Kettler, 2019; Vivian & Falkner, 2019). Teachers need to see the collaborative nature of computing, and practice finding the balance between play and structure, especially when introducing new technology. Learning experiences should both help

teachers see that technology can be used as a support rather than an obstacle to teaching computing, as well as expose teachers to unplugged activities. If technology can bring computing to life for students, it may also bring it to life for teachers.

Summary

The findings of this dissertation provide insight into Minnesota's current computing landscape, particularly highlighting the realities for teachers across multiple contexts. Interestingly, they also illuminate some aspects that have remained with teachers from their PD learnings in the past (e.g., validation/permission, motivation, buy-in, examples, resources, communities, etc.) and disclose where the challenges of integrating computing remain (integration strategies, intended curricula, connective pedagogy, etc.). Most importantly, they reveal some unique implications for PD professionals, MN policymakers, and teacher educators.

Recommendations for Future Research

My dissertation offers suggestions into future research directions that would benefit the field of K-12 CS education. One idea for future research is a deeper exploration of ways to capture and understand teachers' conceptualizations of computing within their classroom as current surveys may not be adequately capturing these differences/nuanced understandings. While surveys like Rich et al.'s (2020) Teacher Beliefs about Coding and Computational Thinking (TBaCCT) and the Computing Attitudes Survey (Leonard et al., 2018; Mouza et al., 2017) provide some insight into how teachers conceptualize CT, they do not capture how teachers envision translating

that understanding into their own classrooms. Building on this work and the surveys created could lead to a survey that better captures those nuances.

A re-imagining or troubling of TPACK-CT could also provide new ways to approach or describe teachers' enacted curriculum of integrating computing. For example, expanding a future iteration of a similar study to this one that includes even more K-12 teachers from multiple contexts and locations (both within and outside of the U.S.) would not only reveal/expand insights, but aid in evolving a shared language to capture teachers' enacted curriculum/their TPACK-CT. Including students in future studies examining teachers' TPACK-CT may also help illuminate more about the enacted curriculum of integrating computing.

Another idea that I am particularly interested in is how teachers are taught to integrate during teacher training programs and PD, and how that impacts, if at all, their approach to integrating computing. Researchers could consider creating research-practice partnerships (RPPs) to better understand how PD and/or teacher education programs can support teachers in deepening their ability and familiarity with integration. An RPP could explore how teachers' confidence in integrating computing shifts/is impacted, if at all, after participating in a PD/class that went beyond only providing examples of an integration lesson and focused on the conceptualization of integration. Using the educational research that exists on integration as a foundation, RPPs could work together to creating learning opportunities that include pedagogical strategies for integration (e.g., what ways can teachers ensure it's not just "cosmetic" (Beane, 1995)), different models of integration so that teachers can name it, and ways in which

technology can support integration. Researchers can explore how familiarity with different ways integration can look, as well as practicing in a learning community the different models of integration, such as looking at content standards alongside CT/CS knowledge, could provide more confidence/understanding for teachers to do this in their own classrooms.

Additional ideas for research include incorporating or switching the focus to students' understanding of CT, the role of technological resources in equitable CT instruction by expanding to even more rural areas, a longitudinal study of teachers' TPACK-CT, the impact of state or national policies on teachers' TPACK-CT as they continue to evolve, and a deeper dive into specific teacher attributes and their relationship with teachers' TPACK-CT.

Limitations

This study had some limitations. While many of the findings align with and support previous literature, the participants were all from Minnesota so caution should be taken in generalizing the dissertation's findings. Minnesota is in the early stages of building K-12 computing education and has taken the approach of weaving standards into existing ones, whereas other states have pursued other approaches to the intended curricula (Guo & Ottenbreit-Leftwich, 2020; MN Department of Education, n.d.). Another limitation is the voluntary nature of the recruitment of participants. For teachers to participate, they had to already have interest/buy-in for computing and feel that they were integrating computing into their classrooms. While the conceptualization and enactment of this varied across the participants (and those who initially expressed interest

during recruitment), their willingness to participate should be a factor to consider - particularly in their CT survey responses and their confidence. As a result of COVID-19 and because of Minnesota's lack of K-12 computing education, the small number of participants is also a limitation. Having at least two participants per grade band or per content area could have provided even more insights into the enacted curriculum of teachers. The impact of COVID-19 is also a contextual factor to consider because it impacted teachers' enacted curriculum as it was the first year back in-person for teachers and students, resulting in the need to navigate new realities and challenges in the classroom that go far beyond just the content/curriculum (Dorn et al., 2021).

In addition, the fact that the study specifically mentioned integration and having some type of PD experiences, meant that these elements would be visible throughout the case portraits, cross-case analysis, and discussion themes. This study also explored TPACK-CT from the teachers' perspectives and did not include students' perspectives/experiences. Including students' voices would provide an even more nuanced understanding of the enacted curriculum of integrating computing in K-12 education. There are also a multitude of possibilities to consider that attributed to the experiences within each case and across cases, including the different previous professional learning experiences each teacher participated in, access to technological resources, their roles and schools, teachers' years of experience and their backgrounds, and the intersectionality of each teacher's individual identities. Including students' experiences and looking at teachers' various attributes also lend themselves to the many directions future research focused on computing integration can take.

In the Beginning/Middle - A Concluding Statement

K-12 computing education research continues to grow and we are in the beginning (or maybe middle depending on your location) stages of its emergence and integration into our U.S. education system. Throughout this study I often called upon how nuanced and complex the art of teaching is, as well as how connected the findings are. My hope is that this study results in a deeper, more nuanced understanding for all of us - the teacher educators, the researchers, the PD professionals, the school administrators, the policymakers, and the teachers themselves. While I do not believe we can quantify a teachers' TPACK-CT nor even clearly define all of the interrelated knowledge domains that teachers pull from while teaching computing, I do believe we can all agree that "it's hard work" (Mr. Pi). So as we ask and call for teachers to continue to do the hard work of integrating computing into their curricula, it is imperative that we provide the supports necessary for them to do the work - from clear intended curricula to professional learning experiences that meet their needs to sharing and evolving both computing curriculum and pedagogical strategies to providing free, accessible technological supports - in order to make computing come alive for both the teachers and the students.

REFERENCES

ACM & IEEE. (2005). *Computing Curricula 2005.* ACM. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2005-march06final.pdf

Alqahtani, M. M., Hall, J. A., Leventhal, M., & Argila, A. N. (2022). Programming in mathematics classrooms: Changes in pre-service teachers' intentions to integrate robots in teaching. *Digital Experiences in Mathematics Education, 8*, 70–98. https://doi.org/10.1007/s40751-021-00096-6

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society, 19*(3), 47–57. https://www.jstor.org/stable/jeductechsoci.19.3.47

Balanskat, A., & Engelhardt, K. (2015). *Computing our future: Computer programming and Coding - Priorities, school curricula and initiatives across Europe*. European Schoolnet. http://www.eun.org/documents/411753/817341/Computing+our+Future_final_2015.pdf/

Barefoot Computing. (n.d.). *Computational thinking concepts and approaches*. Barefoot Computing at School. https://www.barefootcomputing.org/concept-approaches/computational-thinking-concepts-and-approaches

Barksdale, S., Scharber, C., & Chang, Y. (2022). Team Mensa: A case study of supporting middle school girls' interest in computer science through an informal learning program. *Journal of Research on Technology in Education*, 54(3), 359-374. https://doi.org/10.1080/15391523.2020.1864239

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

Beane, J. (1995). Curriculum integration and the disciplines of knowledge. *Phi Delta Kappan, 76*(8), 616-622. http://www.jstor.org/stable/20405413

Bender, E., Hubwieser, P., Schaper, N., Margaritis, M., Berges, M., Ohrndorf, L., Magenheim, J., & Schubert, S. (2015). Towards a competency model for teaching computer science. *Peabody Journal of Education, 90*(4), 519-532. https://doi.org/10.1080/0161956X.2015.1068082

Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in the early childhood classroom.* Routledge.

Blikstein, P. (2018). Pre-college computer science education: A survey of the field. *Google LLC.* https://goo.gl/gmS1Vm

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Computational thinking in compulsory education. Implications for policy and practice. *European Union Scientific and Technical Research Reports*, 1-68. https://doi.org/10.2791/792158

Borowczak, M., & Burrows, A. C. (2019). Ants go marching - Integrating computer science into teacher professional development with NetLogo. *Education Sciences*, *9*(1), 66. https://doi.org/10.3390/educsci9010066

Bower, M., Wood, L. N., Lai, J. W., Howe, C., Lister, R., Mason, R., Highfield, K., & Veal, J. (2017). Improving the computational thinking pedagogical capabilities of school teachers. *Australian Journal of Teacher Education, 42*(3), 53-72. http://dx.doi.org/10.14221/ajte.2017v42n3.4

CBS News Minnesota. (2020, July 27). *Minnesota's schools rank 7th in nation in recent survey*. https://www.cbsnews.com/minnesota/news/minnesotas-schools-rank-7th-in-nation-in-recent-survey/

Codding, D., Alkhateeb, B., Mouza, C., & Pollock, L. (2021). From professional development to pedagogy: Examining how computer science teachers conceptualize and apply culturally responsive pedagogy. *Journal of Technology and Teacher Education, 29*(4), 497-532. https://www.learntechlib.org/primary/p/219931/

Code.org, CSTA, & ECEP Alliance (2022). *2022 State of computer science education: Understanding our national imperative*. https://advocacy.code.org/stateofcs

Coenraad, M., Cabrera, L., Killen, H., Plane, J., & Ketelhut, D. J. (2022). Computational thinking integration in elementary teachers' science lesson plans. *Computational Thinking in PreK-5: Empirical Evidence for Integration and Future Directions*. Association for Computing Machinery, 11–18. https://doi.org/10.1145/3507951.3519283

Corradini, I., Lodi, M., & Nardelli, E. (2017). Conceptions and misconceptions about computational thinking among Italian primary school teachers. In *Proceedings of the 2017 ACM Conference on International Computing Education Research*, 136–144. https://doi.org/10.1145/3105726.3106194

Creswell, J. W. (2013). *Qualitative inquiry & research design* (3rd ed.). Thousand Oaks.

Creswell, J., & Clark, V. (2017). *Designing and conducting mixed methods research* (3rd ed). SAGE Publications.

CSforAllMN. (n.d.). *2023 cs education act*. https://csforallmn.org/cs-ed-bill-2023/

CSforAllMN. (2020a, December). *AP computer science in Minnesota*. https://csforallmn.files.wordpress.com/2021/01/csforall-mn-brief-2-updated-12-8-2020-2.pdf

CSforAllMN. (2020b, December). *Definitions & state overview*. https://csforallmn.files.wordpress.com/2021/01/csforall-mn-brief-1-12-7-20.pdf

CSforAllMN. (2020c, December). *K-12 cs teacher licensure in Minnesota*. https://csforallmn.files.wordpress.com/2021/01/csforall-mn-brief-3-updated-12-7-20.pdf

CSTA. (n.d.). *K-12 cs education glossary*. Computer Science Teachers Association. https://csteachers.org/page/glossary

CSTA, & ISTE. (2011). *Computational thinking. Teacher resources*. https://cdn.iste.org/www-root/2020-10/ISTE_CT_Teacher_Resources_2ed.pdf

Czerkawski, B. (2015). Computational thinking in virtual learning environments. In *Proceedings of E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2015*, 993-997. https://www.learntechlib.org/primary/p/152154/

Delal, H., & Oner, D. (2020). Developing middle school students' computational thinking skills using unplugged computing activities. *Informatics in Education, 19*(1), 1-13. https://infedu.vu.lt/journal/INFEDU/article/22/info

Denning, P. J. (2007). Computing is a natural science. *Communications of the ACM 50*(7), 13–18. https://doi.org/10.1145/1272516.1272529

Denning, P. J., Tedre, M., & Yongpradit, P. (2017). The profession of IT misconceptions about computer science. *Communications of the ACM* 60 (3), 31–33. https://doi.org/10.1145/3041047

Dorn, E., Hancock, B., Sarakatsannis, J., & Viruleg, E. (2021, July). COVID-19 and education: The lingering effects of unfinished learning. *McKinsey & Company*, 1-15. https://www.echs-nm.com/wp-content/uploads/2021/10/covid-19-and-education-the-lingering-effects-of-unfinished-learning-v3.pdf

Duncan, C., Bell, T., & Atlas, J. (2017). What do the teachers think? Introducing computational thinking in the primary school curriculum. In *Proceedings of the Nineteenth Australasian Computing Education Conference (ACE'17)*, 65-74. https://doi.org/10.1145/3013499.3013506

Eichelberger, A., & Leong, P. (2019). Using TPACK as a framework to study the influence of college faculty's beliefs on online teaching. *Educational Media International, 56*(2), 116-133. https://doi.org/10.1080/09523987.2019.1614246

El-Hamamsy, L., Chessel-Lazzarotto, F., Bruno, B., Roy, D., Cahlikova, T., Chevalier, M., Parriaux, G., Pellet, J., Lanarès, J., Zufferey, J. D., & Mondada, F. (2021). A computer science and robotics integration model for primary school: Evaluation of a large-scale in-service k-4 teacher-training program. *Education and Information Technologies 26*, 2445–2475. https://doi.org/10.1007/s10639-020-10355-5

Estapa, A., Hutchison, A., & Nadolny, L. (2018). Recommendations to support computational thinking in the elementary classroom. *Technology and Engineering Teacher, 77*(4), 25. https://www.iteea.org/Publications/Journals/TET/TETDecJan2018.aspx

Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M., & Quille, K. (2019). An international comparison of K-12 computer science education intended and enacted curricula. In *Proceedings of the 19th Koli Calling International Conference on Computing Education Research,* 1-10. https://doi.org/10.1145/3364510.3364517

Fields, D. A., Lui, D., & Kafai, Y. B. (2019). Teaching computational thinking with electronic textiles: Modeling iterative practices and supporting personal projects in *Exploring Computer Science*. In: Kong SC., Abelson H. (eds) *Computational Thinking Education*. Springer. https://doi.org/10.1007/978-981-13-6528-7_16

Flapan, J., Ryoo, J. J., Hadad, R., & Knudson, J. (2021). Preparing school leaders to advance equity in computer science education. *Journal of Computer Science Integration, 4*(1), 1–15. https://doi.org/10.26716/jcsi.2021.10.8.33

Fletcher, C., & Warner, J. (2021). CAPE: A framework for assessing equity throughout the computer science education ecosystem. *Communications of the ACM, 64*(2), 23-25. https://doi.org/10.1145/3442373

Fluck, A., Webb, M., Cox, M., Angeli, C., Malyn-Smith, J., Voogt, J., & Zagami, J. (2016). Arguing for computer science in the school curriculum. *Educational*

Technology & Society, 19 (3), 38–46. https://www.jstor.org/stable/jeductechsoci.19.3.38

Goode, J., Chapman, G., & Margolis, J. (2012). Beyond curriculum: The exploring computer science program. *ACM Inroads, 3*(2), 47-53. https://doi.org/10.1145/2189835.2189851

Goode, J., Flapan, J., & Margolis, J. (2018). Computer science for all: A school reform framework for broadening participation in computer science. In W. G. Tierney, Z. B. Corwin, & A. Ochsner, (Eds.), *Diversifying digital learning: Online literacy and educational opportunity* (pp. 45–65). Johns Hopkins University Press.

Goode, J., Ivey, A., Johnson, S. R. Ryoo, J. J., & Ong, C. (2021) Rac(e)ing to computer science for all: How teachers talk and learn about equity in professional development. *Computer Science Education, 31*(3), 374-399. https://doi.org/10.1080/08993408.2020.1804772

Gretter, S., Yadav, A., Sands, P., & Hambrusch, S. (2019). Equitable learning environments in K-12 computing: Teachers' views on barriers to diversity. *ACM Transactions on Computing Education, 19*(3), 1–16. https://doi.org/10.1145/3282939

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

Guo, M. (2022). *How do K-8 teachers conceptualize and practice teaching computer science?* (Publication No. 29168884) [Doctoral dissertation, Indiana University]. ProQuest Dissertations & Theses Global.

Guo, M., & Ottenbreit-Leftwich, A. (2020). Exploring the K-12 computer science curriculum standards in the U.S. In the *Proceedings of the 15th Workshop on Primary and Secondary Computing Education*, 1-6. https://doi.org/10.1145/3421590.3421594

Guzdial, M. (2019, March). Computing education as a foundation for 21st century literacy. Keynote presentation at the annual meeting of the Special Interest Group on Computer Science Education (SIGCSE) Technical Symposium, Minneapolis, MN. https://doi.org/10.1145/3287324.3290953

Guzdial, M., & Shreiner, T.L. (2021). Integrating computing through task-specific programming for disciplinary relevance: Considerations and examples. In *Computational Thinking in Education: A Pedagogical Perspective*, Aman Yadav and Ulf Dalvad Berthelsen (Eds).

Harry, B., Sturgis, K. M., & Klingner, J. K. (2005). Mapping the process: An exemplar of process and challenge in grounded theory analysis. *Educational Researcher*, *34*(3), 3-13. https://doi.org/10.3102/0013189X034002003

Hatch, J. A. (2002). *Doing qualitative research in education settings*. Suny University Press.

Heafner, T. L. (2018). Elementary ELA/social studies integration: Challenges and limitations. *The Social Studies, 109*(1), 1-12. https://doi.org/10.1080/00377996.2017.1399248

Hinde, E. (2005). Revisiting curriculum integration: A fresh look at an old idea. *Social Studies (Philadelphia, Pa : 1934), 96*(3), 105-111. https://doi.org/10.3200/TSSS.96.3.105-111

Holmes, A. (2020). Researcher positionality - A consideration of its influence and place in qualitative research - A new researcher guide. *Shanlax International Journal of Education, 8*(4), p. 1-10. https://eric.ed.gov/?id=EJ1268044

Hsu, T. C. (2019). A study of the readiness of implementing computational thinking in compulsory education in Taiwan. In: Kong SC., Abelson H. (eds) *Computational Thinking Education*. Springer. https://doi.org/10.1007/978-981-13-6528-7_17

Huang, W., & Looi, C. (2020). A critical review of literature on "unplugged" pedagogies in K-12 computer science and computational thinking education. *Computer Science Education, 31*(1), 83-111. https://doi.org/10.1080/08993408.2020.1789411

Hubbard, A. K. (2017). *Learning to teach computer science: Qualitative insights into secondary teachers' pedagogical content knowledge*. (Publication No. 10602566) [Doctoral dissertation, Northwestern University]. Available from ProQuest Dissertations & Theses Global.

ISTE. (2020). *National educational technology standards for students*. http://www.iste.org

ISTE. (2018, October). *ISTE Announces New Computational Thinking Standards for All Educators*. https://www.iste.org/explore/Press-Releases/ISTE-Announces-New-Computational-Thinking-Standards-for-All-Educators

Jocius, R., O'Byrne, W. I., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2021). Infusing computational thinking into STEM teaching: From professional

development to classroom practice. *Educational Technology & Society, 24* (4), 166–179. https://www.jstor.org/stable/48629253

K-12 Computer Science Framework. (n.d.). *Defining Computer Science*. K12ComputerScience. https://k12cs.org/defining-computer-science/

Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2020). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, *29*(1), 173-187. https://doi.org/10.1007/s10956-019-09798-4

Koehler, M., Mishra, P., & Cain, W. (2013). What is technological pedagogical content knowledge (TPACK)? *The Journal of Education, 193*(3), 13-19. https://doi.org/10.1177/002205741319300303

Kong, S., Lai, M., & Li, Y. (2023). Scaling up a teacher development programme for sustainable computational thinking education: TPACK surveys, concept tests and primary school visits. *Computers and Education*, *194*, 104707, 1-17. https://doi.org/10.1016/j.compedu.2022.104707

Koshy, S., Twarek, B., Bashir, D., Glass, S., Goins, R., Cruz Novohatski, L., & Scott, A. (2022). *Moving towards a vision of equitable computer science: Results of a landscape survey of PreK-12 CS teachers in the United States*. https://landscape.csteachers.org

Kurt, S. (2018, May). TPACK: Technological pedagogical content knowledge framework. *Educational Technology*. https://educationaltechnology.net/technological-pedagogical-content-knowledge-tpack-framework/

Lee, K., Suharwoto, G., Niess, M., & Sadri, P. (2006). Guiding inservice mathematics teachers in developing TPCK (technology pedagogical content knowledge). In C. Crawford, R. Carlsen, K. McFerrin, J. Price, R. Weber & D. Willis (Eds.), *Proceedings of SITE 2006--Society for Information Technology & Teacher Education International Conference*, 3750-3765. https://www.learntechlib.org/primary/p/22684/

Leonard, J., Mitchell, M., Barnes-Johnson, J., Unertl, A., Outka-Hill, J., Robinson, R., & Hester-Croff, C. (2018). Preparing teachers to engage rural students in computational thinking through robotics, game design, and culturally responsive teaching. *Journal of Teacher Education, 69*(4), 386-407. https://doi.org/10.1177/0022487117732317

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Sage.

Loepp, F. (1999). Models of curriculum integration. *The Journal of Technology Studies, 25*(2), 21-25. https://www.jstor.org/stable/43603912

Looi, C. K., How, M.L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education, 28*(3), 255-279. https://doi.org/10.1080/08993408.2018.1533297

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014). Computational thinking in K-9 education. In *Proceedings of the working group reports of the 2014 on innovation & technology in computer science education conference*, 1-29. https://doi.org/10.1145/2713609.2713610

Margot, K., & Kettler, T. (2019). Teachers' perception of STEM integration and education: A systematic literature review. *International Journal of STEM Education, 6*(1), 1-16. https://doi.org/10.1186/s40594-018-0151-2

Mensan, T., Osman, K., & Majid, N. A. A. (2020). Development and validation of unplugged activity of computational thinking in science module to integrate computational thinking in primary science education. *Science Education International, 31*(2), 142-149. https://doi.org/10.33828/sei.v31.i2.2

Merino-Armero, J. M., González-Calero, J. A., Cózar-Gutiérrez, R., & del Olmo-Muñoz, J. (2022). Unplugged activities in cross-curricular teaching: Effect on sixth graders' computational thinking and learning outcomes. *Multimodal Technologies and Interaction, 6*(2), 13. https://doi.org/10.3390/mti6020013

Merriam, S. B., & Tisdell, E. (2016). *Qualitative research: A guide to design and implementation.* Jossey-Bass.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*. Sage.

Miller, D. J. (2020). *Examination of practices used by AP computer science teachers with higher than average female enrollment* (Publication No. 28022033) [Doctoral dissertation, Northern Illinois University]. ProQuest Dissertations & Theses Global.

MN Department of Education. (n.d.). *Computer Science*. https://education.mn.gov/MDE/dse/stds/ComputerScience/

Mohr, K., & Welker, R. (2017). *The role of integrated curriculum in the 21st century school.* (Publication No. 10604000) [Doctoral dissertation, University of Missouri, St. Louis]. ProQuest Dissertations & Theses Global.

Moudgalya, S. K., Yadav, A. Sands, P., Vogel, S., & Zamansky, M. (2021, June). Teacher views on computational thinking as a pathway to computer science. In *26th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2021*), 262-268. https://doi.org/10.1145/3430665.3456334

Mouza, C., Marzocchi, A., Pan, Y. C., & 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. https://doi.org/10.1080/15391523.2016.1146561

Mouza, C., Yadav, A., & Ottenbreit-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. https://www.learntechlib.org/primary/p/184602/

Mouza, C., Yang, H., Pan, Y. C., Yilmaz Ozden, S., & Pollock, L. (2017). Resetting educational technology coursework for pre-service teachers: A computational thinking approach to the development of technological pedagogical content knowledge (TPACK). *Australasian Journal of Educational Technology, 33*(3), 61-76. https://doi.org/10.14742/ajet.3521

Ni, L., & Guzdial, M. (2012). Who AM I?: Understanding high school computer science teachers' professional identity. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, 499-504. https://doi.org/10.1145/2157136.2157283

Nolet, V., & McLaughlin, M. J. (2000). *Accessing the general curriculum: Including students with disabilities in standards-based reform*. Corwin Press, Inc.

Northrup, A. K., Burrows, A. C., & Slater, T. F. (2022). Identifying implementation challenges for a new computer science curriculum in rural western regions of the United States. *Problems of Education in the 21st Century, 80*(2), 353-370. https://eric.ed.gov/?id=EJ1337324

Office of the Legislative Auditor. (2016). *Evaluation Report: Minnesota Teacher Licensure*. State of Minnesota. https://www.auditor.leg.state.mn.us/ped/pedrep/teachers.pdf

Ottenbreit-Leftwich, A., & Yadav, A. (2021). *Computational thinking in prek-5: Empirical evidence for integration and future directions*. ACM and the Robin Hood Learning + Technology Fund.

Ozturk, Z., McMunn Dooley, C. & Welch, M. (2018). Finding the hook: Computer science education in elementary contexts. *Journal of Research on Technology in Education, 50*(2), 149-163. https://doi.org/10.1080/15391523.2018.1431573

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

Parker, W. C. (2005). *Social studies in elementary education*. 12th ed. Pearson Merrill, Prentice-Hall.

Patton, M. Q. (1985, April). Quality in qualitative research: Methodological principles and recent developments. Invited address to Division J of the American Educational Research Association.

Peshkin, A. (1988). In search of subjectivity-one's own. *Educational Researcher, 17*(7), 17-22. https://doi.org/10.3102/0013189X01700701

Peterson, L., Barksdale, S., Scharber, C., Chang, Y., & Freese, J. (2019, April). Computational Thinking in SciGirls Code. Paper presented at American Educational Research Association (AERA) Annual Meeting 2019.

Peterson, L., Barksdale, S., Scharber, C., Schroeder, A., & Wilson Vazquez, A. (2021, May). Skills, equity, and joy in computer science education: Cornerstones for K12 district pathway development. Poster presented at *RESPECT Conference*, Virtual.

Piburn, M., & Sawada, D. (2000). Reformed Teaching Observation Protocol (RTOP) Reference Manual. Technical Report. https://files.eric.ed.gov/fulltext/ED447205.pdf

Remillard, J. T., & Heck, D. J. (2014). Conceptualizing the curriculum enactment process in mathematics education. *ZDM, 46*(5), 705-718. https://doi.org/10.1007/s11858-014-0600-4

Rich, K., & Yadav, A. (2019). Infusing computational thinking instruction into elementary mathematics and science: Patterns of teacher implementation. In *Proceedings of Society for Information Technology & Teacher Education International Conference* 2019. http://ct4edu.org/wp-content/uploads/2019/05/2019_Rich_Yadav_SITE.pdf

Rich, P., Larsen, R., & Mason, S. (2020). Measuring teacher beliefs about coding and computational thinking. *Journal of Research on Technology in Education*, *53*(3), 296-316. https://doi.org/10.1080/15391523.2020.1771232

Ruvalcaba, O., & Rogoff, B. (2022). Children's fluid collaboration versus managing individual agendas: Cultural differences in pair programming. *Journal of Applied Developmental Psychology 81*, 1-12. https://doi.org/10.1016/j.appdev.2022.101438

Sadik, O. (2017). *What do secondary computer science teachers need? Examining curriculum, pedagogy, and contextual support*. (Publication No. 10606747) [Doctoral dissertation, Indiana University]. ProQuest Dissertations & Theses Global.

Sahami, M., Roach, S., Cuadros-Vargas, E., & LeBlanc, R. (2013). *ACM/IEEE-CS computer science curriculum 2013: Reviewing the ironman report*. ACM, 13–14. https://doi.org/10.1145/2445196.2445206

Saldaña, J. (2013). *The coding manual for qualitative researchers*. SAGE Publications.

Sandelowski, M., & Barroso, J. (2002). Finding the findings in qualitative studies. *Journal of Nursing Scholarship, 34*(3), 213-220. https://doi.org/10.1111/j.1547-5069.2002.00213.x

Scharber, C., Peterson, L., Chang, Y., Barksdale, S., & Sivaraj, R. (2021). Critical computing literacy: Possibilities in K-12 computer science education. *Pedagogies: An International Journal*, *16*(2), 136-152. https://doi.org/10.1080/1554480X.2021.1914055

Selby, C., & Woollard, J. (2013). Computational thinking: The developing definition. http://eprints.soton.ac.uk/356481

Shulman, L. S. (1986). Those who understand: Knowledge growth in teaching. Educational Researcher, 15(2), 4-14. https://doi.org/10.3102/0013189X015002004

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142-158. https://doi.org/10.1016/j.edurev.2017.09.003

Stoilescu, D. (2015). A critical examination of the technological pedagogical content knowledge framework. *Journal of Educational Computing Research, 52*(4), 514-547. https://doi.org/10.1177/0735633115572285

Tai, R., Liu, C., Maltese, A., & Fan, X. (2006). Planning early for careers in science. *Science*, *312*(5777), 1143–1144. https://www.science.org/doi/10.1126/science.1128690

Tran, Y. (2019). Computational thinking equity in elementary classrooms: What third-grade students know and can do. *Journal of Educational Computing Research, 57*(1), 3-31.https://doi.org/10.1177/0735633117743918

Tucker, A. (2003). A model curriculum for k-12 computer science: Final report of the ACM k-12 task force curriculum committee. Association for Computing Machinery. https://dl.acm.org/doi/pdf/10.1145/2593247

U.S. Census Bureau. (2021). *QuickFacts Minnesota; United States*. https://www.census.gov/quickfacts/fact/table/MN,US/PST045221

U.S. Department of Education. (n.d.). Institute of Education Sciences, *National Center for Education Statistics*. https://nces.ed.gov

Vakil, S., & de Royston, M. M. (2019). Exploring politicized trust in a racially diverse computer science classroom. *Race Ethnicity and Education, 22*(4), 545-567. https://doi.org/10.1080/13613324.2019.1592846

Veletsianos, G., Beth, B., & Lin, C. (2016). CS teacher experiences with educational technology, problem-based learning, and a CS principles curriculum. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. https://doi.org/10.1145/2839509.2844645

Vee, A. (2018). *Coding literacy*. The MIT Press.

Vivian, R., & Falkner, K. (2019). Identifying teachers' technological pedagogical content knowledge for computer science in the primary years. In *Proceedings of the 2019 ACM International Computing Education Research Conference* (ICER '19). Association for Computing Machinery, 147-155. https://doi.org/10.1145/3291279.3339410

Vogel, S., Santo, R., & Ching, D. (2017). Visions of computer science education: Unpacking arguments for and projected impacts of CS4All initiatives. In the *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. https://doi.org/10.1145/3017680.3017755

Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Educational and Information Technologies, 20*(4), 715-728.

https://link.springer.com/article/10.1007%2Fs10639-015-9412-6

Voogt, J., Fisser, P., Pareja Roblin, N., Tondeur, J., & van Braak, J. (2013). Technological pedagogical content knowledge – a review of the literature. *Journal of Computer Assisted Learning, 29*, 109-121. https://doi.org/10.1111/j.1365-2729.2012.00487.x

Wang, J., Hong, H., Ravitz, J., & Moghadam, S. H. (2016). Landscape of k-12 computer science education in the U.S.: Perceptions, access, and barriers. In P*roceedings of the 47th ACM Technical Symposium on Computing Science Education* (SIGCSE '16). Association for Computing Machinery, 645–650. https://doi-org.ezp3.lib.umn.edu/10.1145/2839509.2844628

Waterman, K. P., Goldsmith, L. & Pasquale, M. (2020). Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology, 29*, 53–64. https://doi.org/10.1007/s10956-019-09801-y

Whiting, K. (2020, October). These are the top 10 job skills of tomorrow - and how long it takes to learn them. *World Economic Forum*. https://www.weforum.org/agenda/2020/10/top-10-work-skills-of-tomorrow-how-long-it-takes-to-learn-them/

Willermark, S. (2018). Technological pedagogical and content knowledge: A review of empirical studies published from 2011 to 2016. *Journal of Educational Computing Research, 56*(3), 315-343. https://doi.org/10.1177/0735633117713114

Willis, G. (2004). *Cognitive interviewing*. Los Angeles: SAGE Publications.

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

Wing, J. M. (2010). Research notebook: Computational thinking - what and why? *The Link.* https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Wing, J. M. (2012, October). Computational thinking. Presentation at the Microsoft Asia Faculty Summit. https://www.microsoft.com/en-us/research/wp-content/uploads/2012/08/Jeannette_Wing.pdf

Woo, A., Lee, S., Tuma, A. P., Kaufman, J.H., Lawrence, R.A., & Reed,R. (2023). *Walking on eggshells—Teachers' responses to classroom limitations on race- or*

gender-related topics: Findings from the 2022 American Instructional Resources Survey. RAND Corporation. https://www.rand.org/pubs/research_reports/RRA134-16.html

Xing, Y., & Zhang, Y. (2020). International comparative study on computational thinking education in K-12. *The Educational Review, 4*(8), 166-175. http://dx.doi.org/10.26855/er.2020.08.002

Yadav, A., Connolly, C., Berges, M., Chytas, C., Franklin, F., Hijón-Neira, R., Macann, V., Margulieux, L., Ottenbreit-Leftwich, A., & Warner, J. R. (2022). A review of international models of computer science teacher education. In *Proceedings of the 2022 Working Group Reports on Innovation and Technology in Computer Science Education (ITiCSE-WGR '22*). Association for Computing Machinery, 65–93. https://doi.org/10.1145/3571785.3574123

Yadav, A. & Heath, M. K. (2022). Breaking the code: Confronting racism in computer science through community, criticality, and citizenship. *TechTrends*, *66*(3), 450-458. https://doi.org/10.1007/s11528-022-00734-9

Yadav, A., Krist, C., Good, J., & Caeli, E. N. (2018). Computational thinking in elementary classrooms: Measuring teacher understanding of computational ideas for teaching science. *Computer Science Education, 28*(4), 371-400. https://doi.org/10.1080/08993408.2018.1560550

Yadav, A., Larimore, R., Rich, K., & Schwarz, C. (2019). Integrating computational thinking in elementary classrooms: Introducing a toolkit to support teachers. In *Proceedings of Society for Information Technology & Teacher Education International Conference 2019*, 1-4. http://ct4edu.org/wp-content/uploads/2019/05/2019_Yadav_SITE.pdf

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education 14*(1). http://dx.doi.org/10.1145/2576872

Zha, S., Jin, Y., Moore, P., & Gaston, J. (2020). Hopscotch into coding: Introducing pre-service teachers computational thinking. *TechTrends, 64*(1), 17-28. https://link.springer.com/article/10.1007/s11528-019-00423-0

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), 1-18. https://doi.org/10.1145/3410631

Zhu, M., & Wang, C. (2023). K-12 computer science teaching strategies, challenges, and teachers' professional development opportunities and needs. *Computers in the Schools*, 1-22. https://doi.org/10.1080/07380569.2023.2178868

APPENDICES

Appendix A

IRB Approval

UNIVERSITY OF MINNESOTA

Twin Cities Campus

Human Research Protection Program
Office of the Vice President for Research

Room 350-2
McNamara Alumni Center
200 Oak Street S.E.
Minneapolis, MN 55455

612-626-5654
irb@umn.edu
https://research.umn.edu/units/irb

EXEMPTION DETERMINATION

February 9, 2021

Cassandra Scharber

612-625-6607
scharber@umn.edu

Dear Cassandra Scharber:

On 2/9/2021, the IRB reviewed the following submission:

Type of Review:	Initial Study
Title of Study:	Teaching Computational Thinking in the Elementary Classroom
Investigator:	Cassandra Scharber
IRB ID:	STUDY00011936
Sponsored Funding:	None
Grant ID/Con Number:	None
Internal UMN Funding:	None
Fund Management Outside University:	None
IND, IDE, or HDE:	None
Documents Reviewed with this Submission:	• Recruitment Email.docx, Category: Recruitment Materials; • Teacher Reflections Protocol.docx, Category: Other; • Survey Template.docx, Category: Other; • Observation Protocols.docx, Category: Other; • Consent Form.docx, Category: Consent Form; • Letters of School Support.docx, Category: Letters of Support / Approvals (Location); • Barksdale_Teaching_CT_ HRP-580 (2).docx, Category: IRB Protocol; • Pre-Recruitment Email.docx, Category: Other;

	• Semi-Structured Interview Questions.docx, Category: Other; • Interested Participant Form.docx, Category: Other;

The IRB determined that this study meets the criteria for exemption from IRB review. To arrive at this determination, the IRB used "WORKSHEET: Exemption (HRP-312)." If you have any questions about this determination, please review that Worksheet in the HRPP Toolkit Library and contact the IRB office if needed.

The IRB understands that as long as the pre-recruitment strategies are preparatory to formal research contact and are just generically asking teachers if they are interested in the research this is acceptable.

The IRB notes that approval is granted upon the condition that the researcher submits letters of approval from the schools and the school district where the proposed research will be conducted. The IRB notes that no recruitment of study subjects or collection of research data can begin until these letters are obtained and submitted with a Modification in ETHOS for review.

This study met the following categories for exemption:

• (1) Research conducted in established or commonly accepted educational settings, involving normal educational practices. (Both the procedures involve normal education practices and the objectives of the research involve normal educational practices)

• (2) Research that only includes interactions involving educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures, or observation of public behavior (including visual or auditory recording) if at least one of the following criteria is met: (ii) Any disclosure of the human subjects' responses outside the research would not reasonably place the subjects at risk of criminal or civil liability or be damaging to the subjects' financial standing, employability, educational advancement, or reputation

Ongoing IRB review and approval for this study is not required; however, this determination applies only to the activities described in the IRB submission and does not apply should any changes be made. If changes are made and there are questions about whether these activities impact the exempt determination, please submit a Modification to the IRB for a determination.

In conducting this study, you are required to follow the requirements listed in the Investigator Manual (HRP-103), which can be found by navigating to the HRPP Toolkit Library on the IRB website.

For grant certification purposes, you will need these dates and the Assurance of Compliance number which is FWA00000312 (Fairview Health Systems Research FWA00000325, Gillette Children's Specialty Healthcare FWA00004003).

Appendix B

Recruitment Email

Subject: Paid research opportunity through the University of Minnesota!

Hello there!

Are you a K-12 school teacher in Minnesota? Do you have knowledge of computational thinking (CT)? Are you interested in improving our state's participation in and opportunities for computer science (CS) experiences so that they are more equitable? If so, I may have a great opportunity for you beginning in Fall 2021 - **and it's paid**! If you are interested or want more information, I'd love for you to fill out this **short survey**!

My name is Sarah Barksdale, and I am a graduate student working on my dissertation to complete my PhD at the University of Minnesota. I have asked _____ to share this email with you because of your experience and knowledge of computer science. I am seeking interested K-12 educators in Minnesota who have participated in some sort of computer science training or professional development (PD) in the last few years who are working to integrate CT/CS into their curriculum. My research works to better understand how educators approach integrating computer science after they have participated in professional learning experiences. Findings from this study will be helpful to identify problematic areas of current training approaches, notice key supports to guide future resource development, and create more alignment between the professional learning experience and what happens in the classroom.

This research will not start until the **upcoming school year (Fall 2021)**, but I would love to connect now. Based on your responses and interest, a small pool of participants will be invited to participate in the study. All selected research participants will receive **at least a $100 gift card** for participating in the research study in 2021-2022. **If you are interested or want more information, please complete this short survey and I will be in touch with you soon**! *Filling out this survey does NOT obligate you to participate or automatically sign you up to be in the study.*

What will it mean to participate in the study during 2021-2022?

- Completing a short online survey at the beginning of our study (Fall 2021)

- Allow me to observe you in your classroom (either virtually or face to face depending on COVID-19 restrictions) up to three times when you are teaching a lesson that integrates CT (you choose when)
- Participate in 2 interviews: one after completing the survey (Fall 2021) and one at the end of the research study
- Completing a brief reflection after your lesson (either written or verbal)
- Provide access to curriculum and documents you create to teach your lessons if applicable

Additional details if you choose to participate in 2021:

Your name will remain private. The researchers are the only people who will know your name. Any publications or reports will not have your name, as we will use a pseudonym. The consent sheet will outline some other procedures we take to ensure your privacy and confidentiality.

Your decision whether you participate in this study will not change how the researchers treat you or your relationship with _____ It will also not affect current or future relations with the researcher or the University of Minnesota. If you decide to participate you will be free to drop out of the study at any time without affecting those relationships.

Please reach out to me with any questions or concerns.

Appendix C

Recruitment Interest Survey

Teacher Recruitment for 2021-2022

Start of Block: Introduction

Hello!
My name is Sarah Barksdale, and I am a graduate student working on my dissertation at the University of Minnesota. I am seeking interested K-12 educators in Minnesota who have participated in some sort of computer science (CS) learning experience, training, or professional development (PD) in the last few years who are working to integrate CS into their curriculum. The purpose of my research is to better understand how educators approach integrating CS after they have participated in some type of professional learning experience. Findings from this study will be helpful to identify problematic areas of current training approaches, notice key supports to guide future resource development, and create more alignment between the PD experience and what happens in the classroom.

This research will take place during this 2021-2022 school year. Based on your responses and interest collected in this form, a small pool of participants will be invited to participate in the study. All chosen research participants will receive at least a $100 Amazon gift card for participating in the research study in 2021-2022. If you are interested or want more information, please complete this short survey and I will be in touch with you soon! Filling out this survey does NOT obligate you to participate or automatically sign you up to be in the study.

Please reach out to Sarah Barksdale [redacted]) with any questions or concerns!
Thank you for your consideration, Sarah Barksdale

End of Block: Introduction

Start of Block: Details

Q2 First and Last Name

_____

Q3 Preferred email address

_____

Q4 Phone number

_____

Q5 Please select the grade(s) you are teaching during the 2021-2022 school year.

- [ ] Kindergarten (1)
- [ ] 1st grade (2)
- [ ] 2nd grade (3)
- [ ] 3rd grade (4)
- [ ] 4th grade (5)
- [ ] 5th grade (6)
- [ ] 6th grade (7)
- [ ] 7th grade (8)
- [ ] 8th grade (9)
- [ ] 9th grade (10)
- [ ] 10th grade (11)
- [ ] 11th grade (12)
- [ ] 12th grade (13)

Q6 What content area(s) are you teaching this school year? (E.g., elementary education, music, art, science, math, etc.)

_____

Q7 Please share the names of your school and district.

_____

Q8 Have you participated in any professional learning(s) related to computer science? (e.g., courses, workshops, webinars, cohorts, etc.)

- ○ Yes (1)
- ○ No (2)

Display This Question:

If Have you participated in any professional learning(s) related to computer science? (e.g., courses... = Yes

Q9 Can you please share more about the professional learning you participated in, including:the name and website if applicable when and length of training the format of the training (e.g., in person, webinar, school provided, etc.)

_____

Q11 Please share a brief example of a lesson you teach that integrates computer science into other content.

_____

Q12 Do you have any questions about this study?

_____

Appendix D

TPACK-CT Observation Protocol

Technological Knowledge - CT		
Guiding Question(s)	Examples	Notes
Were computing tools, terminology, practices, and/or dispositions used for cultivating understanding of CT?		
Content Knowledge-CT		
Guiding Question(s)	Examples	Notes
Were connections made between CT and existing content?		
Explicit or implicit connections?		
Pedagogical Knowledge-CT		
Guiding Question(s)	Examples	Notes
What pedagogical methods were used to make connections between content & CT?		
How was the lesson structured to make connections between content and CT?		
Other Noticings		
Context/Classroom	CT Characteristics & Dispositions Visible	Notes

Appendix E

CT Beginning Survey

Introduction:
Hello!
Thank you so much for participating in my research study focused on computer science/computational thinking in the classroom.

This survey will take about 10-15 minutes to complete and will ask you about your teaching experience(s), as well as some of your beliefs around computing and education. There are no wrong or right answers, and what you share here will be used to guide our first interview. During that interview, you will also be able to clarify your thoughts if you want to explain something further.

The information you provide on this survey will ONLY be used for research purposes, and will NOT be shared with your district or school administration. All identifiable information will be confidential.

I so appreciate and value your time, insight, and experience!

Demographics:

- Name
- Current teaching role(s), grade level(s), and/or content area(s) for the 2021-2022 school year.
- How long teaching:
    - Overall
    - This grade(s)
    - CT concepts
- How often do you teach computing &/or coding activities? [fixed selections]
    - 1 lesson/year
    - 2+ lessons/year
    - 1x month
    - 1x week
    - 2-3 times/week
    - Every day
    - Other: _____
- On average, how long are your computer science/computational thinking lessons?
    - 15 mins or less
    - 16-30 mins
    - 31-45 mins
    - 46+ min

The following items used a 4 point Likert-scale for respondents to select if they 1-strongly disagree, 2-disagree, 3-agree, & 4-strongly agree for each of the following statements:

Please indicate the extent to which you (dis)agree with each of the following statements about computational thinking.

- Computational thinking is understanding how computers work.
- Computational thinking involves thinking logically to solve problems.
- Computational thinking involves using computers to solve problems.
- Computational thinking involves abstracting general principles and applying them to other situations.

Note: The following questions ask about your perceptions of computational thinking (CT) and teaching kids CT. Some of the following terms are used throughout:

- (Computer) programming: The act of developing instructions that a computer will read and execute. Programming may be expanded to include devices other than traditional computers (e.g., robots, chips, small equipment)
- Coding: A synonym often used in place of "(computer) programming"
- Computing: Encompasses coding, but also refers to the thinking processes involved in solving problems with the use of machines

The following items used a 4 point Likert-scale for respondents to select if they 1-strongly disagree, 2-disagree, 3-agree, & 4-strongly agree for each of the following statements:

Please indicate the extent to which you (dis)agree with each of the following statements about your confidence with coding.

- I can describe fundamental computing concepts (e.g. loops, variables, algorithms, conditional logic).
- I am NOT comfortable with learning computing concepts.
- I can correct mistakes in the coding of a computer program on my own.
- I can look at a computer program and explain the purpose of each command.
- I am good at finding patterns in data.
- I can apply Boolean logic (e.g., IF, AND, NOT, OR) to solve problems with multiple conditions.
- I can read a formula (e.g., algorithm, equation, input/output process) and explain what it should do.

Please indicate the extent to which you (dis)agree with each of the following statements about teaching coding.

- I can integrate computing into my current curriculum.
- I believe that I have the requisite computer programming skills to integrate computing programming into my class lessons.
- I can recognize and appreciate computing concepts in all subject areas.
- I can explain computing concepts well enough to be effective in teaching computing.
- I can explain how computing concepts are connected to daily life.
- I can develop and teach effective computing lessons.

Please indicate the extent to which you (dis)agree with each of the following statements.

- Computing should be taught in elementary school.
- Learning about computing can help elementary students become more engaged in school.
- My current teaching situation does NOT lend itself to teaching computing concepts to my students.
- Knowledge of computer programming is NOT needed in most careers.
- Computing is an important 21st-century literacy.

Open-ended questions:

- In your own words, how do you define the term computational thinking?
- How can technologies be used to support the development of students' computational thinking skills, if at all?
- What teaching strategies do you, or could you, use to support students learning computational thinking?
- If you have something else you would like to share, you can use this space to do so.

Appendix F

The Computing Attitudes Surveys Used for Building This Study's Survey
Surveys used a 4 point Likert-scale for respondents to select if they 1-strongly disagree, 2-disagree, 3-agree, & 4-strongly agree

Yadav et al., 2014's original survey:	Modified Computational Thinking Survey (Leonard et al., 2018)
<u>Definition</u> 1. Computational thinking is understanding how computers work 2. Computational thinking involves thinking logically to solve problems 3. Computational thinking involves using computers to solve problems 4. Computational thinking involves abstracting general principles and applying them to other situations <u>Comfort</u> 5. I do not think it is possible to apply computing knowledge to solve other problems 6. I am not comfortable with learning computing concepts 7. I can achieve good grades (C or better) in computing courses 8. I can learn to understand computing concepts 9. I do not use computing skills in my daily life 10. I doubt that I have the skills to solve problems by using computer applications <u>Interest</u> 11. I think computer science is boring 12. The challenge of solving problems using computer science appeals to me 13. I think computer science is interesting 14. I will voluntarily take computing	1. Computational thinking is understanding how computers work. 2. Computational thinking involves thinking logically to solve problems. 3. Computational thinking involves using computers to solve problems. 4. Computational thinking involves abstracting general principles and applying them to other solutions. 5. I do not think it is possible to apply computing knowledge to solve other problems. 6. I am not comfortable with learning computing concepts. 7. I can achieve good grades (C or better) in computer courses. 8. I can learn to understand computing concepts. 9. I use computing skills in my daily life. 10. I doubt that I have the skills to solve problems by using computer applications. 11. I think computer science is boring. 12. The challenge of solving problems using computer science appeals to me. 13. I think computer science is interesting. 14. I will voluntarily take computing courses if I were given the opportunity. 15. Computational thinking can be incorporated in the classroom by using computers in the lesson plan. 16. Computational thinking can be

<table>
<tr>
<td>courses if I were given the opportunity

<u>Classroom</u>
15. Computational thinking can be incorporated in the classroom by using computers in the lesson plan
16. Computational thinking can be incorporated in the classroom by allowing students to problem solve

<u>Career</u>
17. Knowledge of computing will allow me to secure a better job
18. My career goals do not require that I learn computing skills
19. I expect that learning computing skills will help me to achieve my career goals
20. I hope that my future career will require the use of computing concepts
21. Having background knowledge and understanding of computer science is valuable in and of itself</td>
<td>incorporated in the classroom by allowing students to problem solve.
17. Knowledge of computing will allow me to improve my performance in my career.
18. My career does not require that I learn computing skills.
19. I expect that learning computing skills will help me to achieve my career goals.
20. I hope that as my career continues it will require the use of computing concepts.
21. Having background knowledge and understanding of computer science is valuable in and of itself.</td>
</tr>
<tr>
<td colspan="2">Open-ended questions from Mouza et al., 2017:

1. What do you think the term computational thinking means?
2. How can technologies be used to support the development of students' CT skills?</td>
</tr>
</table>

Appendix G

Semi-Structured Interview Question Examples

Context:

1. Share with me a little about yourself and your educational journey - how did you get to where you are?
2. Tell me about your current role and your current school.
3. What types of resources does your school have? Technology?
4. Support in general? Support in teaching computing?
5. Are there other teachers teaching or integrating CS or CT?
6. Do you think there is demand in your district for CS? From who? (Parents, students, teachers, admin, etc.)

CK:

7. Can you share with me when you first became familiar with computer science or computational thinking?
8. Why begin teaching CT?
9. I believe in the first survey, you mentioned having PD experiences - are there go to resources you use?
10. How much do you feel teachers have to know to be able to teach CS/CT?
11. What is your approach to bringing in CS/CT? (i.e., stand alone, integrating, based off of PD ideas, etc.)
12. What content areas, if any, do you integrate it into? Easy/hard?
13. Do you feel you integrate it well or are prepared to?
14. What helps/makes it harder?
15. You mentioned [insert survey/prior convo, etc. info] can you tell me more about that? Any collaborating with science teacher? Why science?
16. Where do you get your curriculum or how do you develop it?
17. Any integration or working with other teachers? Themes?

T:

18. You mentioned [using student centered technologies for teaching computing] can you give some specific examples of what you use?
19. How do you choose what to use and when?
20. Do you feel like technology is a barrier/resource, if at all?

P:

21. How often do you get to teach the same group of students CS/CT?
22. Do you set learning or lesson objectives around CS/CT?

23. Can you give an example of a [insert previous info shared] you've done or plan to do?

Appendix H

Guided Reflection Draft

Using the questions below as a guide, please take 5-10 minutes to reflect on how your lesson went today. You may audio record your reflection or write it out.

- What were the goals of your lesson today? What did you want the students to leave knowing?
- How did you decide to incorporate computing into today's lesson? How did you feel it enhanced, if at all, the content?
- How explicit do you think CT/CS was?
- Were there any strategies you planned to use ahead of time? Why or why not?
- How do you think students responded to today's lesson?
- Is there anything you would do differently?
- How, if at all, do you think today's lesson supported students' development of CT/CS skills? Which skills specifically?
- What is your personal take-away from today's lesson?

Appendix I

Cross-Case Word Clouds Excerpt

Codes related to TPK, TK-CT, & TCK-CT example: