

2022

COMPUTATIONAL THINKING IN ELEMENTARY SCIENCE

Jennifer Pietros
University of Rhode Island, pietros@uri.edu

Follow this and additional works at: https://digitalcommons.uri.edu/oa_diss

Recommended Citation

Pietros, Jennifer, "COMPUTATIONAL THINKING IN ELEMENTARY SCIENCE" (2022). *Open Access Dissertations*. Paper 1354.
https://digitalcommons.uri.edu/oa_diss/1354

This Dissertation is brought to you for free and open access by DigitalCommons@URI. It has been accepted for inclusion in Open Access Dissertations by an authorized administrator of DigitalCommons@URI. For more information, please contact digitalcommons-group@uri.edu.

COMPUTATIONAL THINKING IN ELEMENTARY SCIENCE

By

JENNIFER PIETROS

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

IN

EDUCATION

UNIVERSITY OF RHODE ISLAND

AND

RHODE ISLAND COLLEGE

2022

DOCTOR OF PHILOSOPHY DISSERTATION
OF
JENNIFER PIETROS

APPROVED:

Dissertation Committee:

Major Professor

Minsuk Shim

Sara Sweetman

Anne Goodrow

Sally Hamouda

RIC:

Jeannine Dingus-Eason
Dean, Feinstein School of Education – RIC

URI:

Brenton DeBoef
Dean, The Graduate School - URI

UNIVERSITY OF RHODE ISLAND
AND
RHODE ISLAND COLLEGE
2022

ABSTRACT

This dissertation presents exploratory research on computational thinking in elementary science classrooms. The research presented here attempts to bring knowledge and awareness of computational thinking in elementary science to elementary teachers, administrators, curriculum developers, researchers, and policy makers by learning about descriptions of CT and the different concepts and approaches involved with it. In addition, this research aims to gain a better understanding of the computational thinking concepts and approaches that are occurring in science classrooms across a northeast state and the different variables that predict integration within instruction. This information will be used to determine the areas of need for future implementation efforts.

Through a close examination of literature on computational thinking integrated in elementary science, survey data, and analysis of work performed by a Design-Based Implementation Research (DBIR) Team this dissertation adds to a growing body of literature regarding the integration of CT into elementary science. Chapter 1 is an introduction to the dissertation. Chapters 2, 3, and 4 are written in manuscript format, and are titled as follows: Chapter 2 is entitled “How is Computational Thinking Defined in Elementary Science?” Chapter 3 is entitled “Predicting Computational Thinking in Elementary Science using a Multilevel Model Approach.” Chapter 4 is entitled “Teacher Practices for CT Implementation in Elementary Science.” Chapter 5 is the conclusion of the major findings from this research, policy implications, limitations, and future recommendations.

Three of the chapters in this dissertation are meant to serve as stand-alone, publishable manuscripts. Each chapter uses a different methodology and analytical framework and offers its own distinct findings. At the same time, each individual manuscript is intended to build upon and is informed by findings from the other manuscripts. Policy implications across the three manuscripts and recommendations for future study are shared.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my major professor, Minsuk Shim, for advising me during my time as a doctoral student. Your support, encouragement, and guidance throughout this process has been exceptional. I cannot thank you enough for taking the time to meet with me often and explain statistics and how to organize my work with such patience. The sharing of your time, knowledge, and experience has helped me achieve completing this dissertation. I appreciate your help so much and am incredibly grateful.

I also cannot thank Sara Sweetman enough for giving me the opportunity of a lifetime to join her research team. You have been an amazing mentor and have taught me so much about the research process from creating surveys, going through the IRB process, collecting, and analyzing data, presenting at conferences, writing for publications, and writing proposals for future research. You have truly given me an outstanding learning experience. I have learned so much from you and am forever grateful.

I would also like to thank my other committee members, Anne Goodrow and Sally Hamouda for being so supportive in the research process and offering valuable feedback and advice to help me be successful. I also would like to thank my former advisor, Michael Rice, from the Aquaculture Department, for being the chair of my dissertation. I had such a wonderful learning experience twenty years ago that I was glad to come back to URI when I started the doctoral program.

I would also like to thank all the professors from the URI/RIC doctoral program who taught me throughout the program. You pushed my thinking and helped me see the world in a different way which has motivated and inspired me to be a better teacher. I

also want to thank the 2016 Cohort for the great discussions and diverse perspectives. I learned a lot from each of you. A special thank you to Safie Sagna and Shawna Iulu who have been a great support system throughout the process. Who knew our qualitative observations at the Cheesecake Factory would turn into such a wonderful friendship?

Professionally, I would like to thank all the teachers, researchers, and STEM+C individuals involved with the DBIR research team. We accomplished a lot together and I learned so much from such a fun diverse group of people. I also want to thank all the teachers from Feinstein Middle School in Coventry, especially my team and the sixth-grade science teachers. You all have been so encouraging throughout this process, especially during some of the toughest years teaching during this pandemic. You are some of the hardest working people I know!

To my sisters and brother (Michelle, Marcy, and Andy), family, and friends, thank you for supporting me in so many ways. I would not be where I am today without you. I also want to give a special thank you to my first teachers in life, my mom and dad. You have taught me to persevere when times get tough. You also have passed down your love for learning and love of science to me. I am forever grateful for this gift and hope I pass it down to my children and grandchildren along with one of my mom's favorite sayings, "Nobody can ever take an education away from you. It is yours to keep."

Finally, I want to thank my husband, Pete, for your support, encouragement, and love you gave while I went through this process. You have taken on a lot of the household responsibilities throughout the dissertation process. I appreciate it more than you will ever know. Of course, your work would not be complete without having to endure shoveling a ton of snow during a blizzard while I finished writing this

dissertation. Thank you! I love you, and I could not have accomplished this PhD without your help!

PREFACE

A manuscript format was used in the preparation of this doctoral dissertation. Three separate manuscripts were written for publication. Manuscript 1 is a literature review focusing on how computational thinking is defined in elementary science and the steps a Design-Based Implementation Research Team took to define computational thinking in teacher-friendly language. Manuscript 2 focuses on the teacher and district factors that predict the use of computational thinking concepts and approaches by analyzing cross-sectional survey data using a multilevel model analysis. Manuscript 3 investigates the types of lessons that include opportunities for teaching computational thinking concepts and approaches and examples of what these lessons look like. The papers are written in a manuscript format for journal submission as cited below:

MANUSCRIPT 1: “How is Computational Thinking Defined in Elementary Science?” is currently in preparation for submission to *Tech Trends*. Some of the ideas from this paper will be presented as a poster at the SIGCSE 2022 Conference, March 3–5, 2022, Providence, RI, USA. Association of Computing Machinery. DOI: <https://doi.org/10.1145/3478432.3499120>

MANUSCRIPT 2: “Predicting Computational Thinking in Elementary Science Using a Multilevel Model Approach” is currently in preparation for submission to the *Journal of Science Education and Technology*.

MANUSCRIPT 3: “What Does Computational Thinking Look Like in Elementary Science” is currently in preparation for submission to the *Elementary School Journal*.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	ii
ABSTRACT.....	iii
ACKNOWLEDGMENTS	iv
PREFACE.....	vii
LIST OF TABLES.....	xi
LIST OF FIGURES	xii
TABLE OF CONTENTS	xiii
<u>CHAPTER 1:</u> INTRODUCTION TO COMPUTATIONAL THINKING IN ELEMENTARY SCIENCE.....	1
Introduction and Statement of the Problem	2
How is Computational Thinking defined in elementary science?	5
Teacher practices for CT implementation in elementary science.....	11
Summary of Three Paper Manuscript.....	13
<u>CHAPTER 2:</u> HOW IS COMPUTATIONAL THINKING DEFINED IN ELEMENTARY SCIENCE?	15
Introduction.....	17
Conceptual Framework of TPACK	20
Computational Thinking Background.....	21
Literature Review of CT in K–5 Elementary Science	26
Procedure	45
Crosswalk of the standards	46
Creating a Padlet.....	47
Analysis of survey data.....	50
Value-mapping.....	53
Brainstorm for shared definition.....	55
Discussion.....	56
References.....	60
<u>Appendix A:</u> References Used in Literature Review.....	65
<u>Appendix B:</u> Brainstorm of What CT Is and Is Not	68
<u>CHAPTER 3:</u> PREDICTING COMPUTATIONAL THINKING IN ELEMNTARY SCIENCE USING A MULTILEVEL MODEL APPROACH.....	69
Abstract.....	70

Introduction and Statement of the Problem	71
Theoretical and Conceptual Frameworks	73
Literature Review.....	78
What is Computational Thinking?	78
CT in elementary science.....	79
Implementation challenges	81
CT curriculum.....	82
Teacher development	84
Research Design.....	87
Methods.....	88
Participants.....	88
Survey	90
Variables of interest	93
Results.....	94
Discussion.....	111
Limitations	115
Implications and recommendations	116
References.....	118
<u>CHAPTER 4: TEACHER PRACTICES FOR COMPUTATIONAL THINKING</u>	
IMPLEMENTATION IN ELEMENTARY SCIENCE.....	126
Abstract.....	127
Introduction.....	128
Theoretical Framework.....	131
Purpose of integrating CT into elementary science	132
Policies for Integrating CT.....	134
Programs for Integrating CT	137
Practices for Integrating CT in Science	139
Methods.....	141
Participants.....	142
CT survey.....	142
Analysis.....	145
Results.....	146
Summary of Findings and Policy Implications.....	155
Limitations and suggestions.....	159
References.....	161

Appendix A.....	167
CHAPTER 5: CONCLUSION OF COMPUTATIONAL THINKING IN ELEMENTARY SCIENCE.....	170
Summary of Findings.....	172
Consensus on CT	176
Teacher and District Level Factors Impacting CT Integration	177
Elementary Science Instructional Time	182
Opportunities for CT Integration	183
Limitations and Recommendations for Future Research.....	185
Conclusion	186
BIBLIOGRAPHY	188
Appendix A: Computational Thinking in Elementary School Classrooms	202

LIST OF TABLES

CHAPTER 2

Table 1. Research Studies on CT Integrated in Elementary Science.....	30
Table 2. Teacher-Friendly Definitions for CT.....	49
Table 3. Total CT Concept Examples from CT Survey.....	52
Table 4. Total CT Approaches Examples from CT Survey.....	52
Table 5. Teacher-Friendly Definition of CT for Elementary Level.....	55

CHAPTER 3

Table 1. Teacher and District Level Variables for CT Frequencies.....	93
Table 2. HLM Models for Algorithms.....	102
Table 3. HLM Models for Decomposition.....	103
Table 4. HLM Models for Patterns.....	104
Table 5. HLM Models for Abstraction.....	105
Table 6. HLM Models for Tinkering.....	106
Table 7. HLM Models for Creating.....	107
Table 8. HLM Models for Debugging.....	108
Table 9. HLM Models for Perseverance.....	109
Table 10. HLM Models for Collaboration.....	110

CHAPTER 4

Table 1. Example of CT Definition	146
Table 2. Examples of CT Concepts and Approaches	152
Table 3. Accurate and Inaccurate Examples of CT Concepts and Approaches...	153

LIST OF FIGURES

CHAPTER 2

Figure 1. Frequency of CT Components Used in Research Studies.	42
Figure 2. CT Research Studies Conducted in K–5 Elementary Science.....	43
Figure 3. Padlet Created by DBIR Group.....	48
Figure 4. Poster Created with Teacher-Friendly Definitions.....	50
Figure 5. Value-Map of Wordles Created at Beginning of Research.....	54
Figure 6. Value-Map of Wordles Created at Middle of Research.....	54
Figure 7. Value-Map of Wordles Created at End of Research.....	54

CHAPTER 3

Figure 1. CT-TPACK Model.....	76
Figure 2. Example of CT Concept from Survey.....	91
Figure 3. Average Minutes per Week Teaching CT Concepts and Approaches...	95

CHAPTER 4

Figure 1. Example of Survey Question.....	144
Figure 2. Time Spent Teaching Science.....	147
Figure 3. Percentage of Lessons Including CT Concepts and Approaches.....	148

CHAPTER 1

INTRODUCTION TO COMPUTATIONAL THINKING IN ELEMENTARY SCIENCE

Introduction and Statement of the Problem

Children are growing up in a world where they are surrounded by computers and technology in almost all aspects of their lives. As computers become more pervasive in their everyday life, a high-quality science, technology, engineering, and math (STEM) education, with a focus on computational thinking skills, is needed to collect, analyze, and use data to solve complex problems such as climate change, poverty, diseases, shortage of resources, and biodiversity losses (Mills et al., 2021; Rischard, 2007). STEM related employment in the United States, which includes computer science, is growing at a faster pace than non-STEM occupations (Noonan, 2017; White & Shakibnia, 2019). With this demand there is a need to increase access to computer science subject matter for every child because not only does it address the needs of the workforce and skills needed in the digital age, but more importantly it addresses foundational educational needs such as being able to think critically to solve problems, data analysis, and modeling skills (Papert, 1980; Khine, 2018). By increasing access to computer science by all children, underrepresented groups and students from urban, rural, and low socioeconomic areas will have more opportunities to participate in this rapidly expanding field (Code.org 2021).

One key component being addressed in computer science is computational thinking (CT). This process was introduced by Jeannette Wing in 2006 in her famous article where she described CT as “a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child’s analytic ability” (Wing, 2006, p. 33). The Next Generation Science Standards states computational thinking “involves strategies for organizing and

searching data, creating sequences of steps called algorithms, and using and developing new simulations of natural and designed systems” (NGSS, 2013, Appendix F, p. 10). There are many definitions for this term and its component parts found throughout the literature but there is no agreed upon definition which makes it difficult for implementation. However, there is consensus that these are skills all children should be taught in their early years of education and is a skill worth implementing into current instruction. One of the best ways to get children motivated in developing computer science skills is to engage and expose them to CT at an early age. Implementation of CT at the elementary level needs to be thoughtful and done with the needs of districts and teachers in mind for engagement to be effective. The problem is school systems need resources on how to effectively integrate the concepts and approaches into their epistemology, content, pedagogy, and practice. Currently, there is not a set curriculum implemented in most school systems or best practices established to educate teachers on how to do this effectively. In order to make this happen, systematic research is needed before valuable implementation can take place and be sustainable.

The goal of the three research papers in this dissertation is to bring knowledge and awareness of computational thinking in elementary science to elementary teachers, administrators, curriculum developers, researchers, and policy makers by learning about descriptions of CT and the different concepts and approaches involved with it. In addition, this research aims to gain a better understanding of the computational thinking concepts and approaches that are occurring in science classrooms across a northeast state and the different variables that predict integration within instruction. This information will be used to determine areas of need for implementation.

In order to integrate CT into elementary science classrooms, an overarching theoretical framework for this dissertation is Bybee's (2013) 4P model. This model is composed of four dimensions which are Purpose, Policy, Program, and Practice. The 4P model has been used in STEM reform efforts and is being used to guide implementation efforts for integrating CT in science instruction. The *Purpose* is the goal towards which researchers, practitioners, administrators, policymakers, parents, and students strive. A goal of this research is to explore and find teacher strategies that will help with integrating CT into lessons while enhancing student learning of both science and computer science. Manuscript 1 in this dissertation addresses the purpose of what CT is and the component parts. *Policies* are concrete translations of the purpose. Frameworks and standards have been put in place, but execution still needs to happen. Manuscript 2 discusses some of the policies that have been put into place for CT integration. *Program* is the curriculum materials that embody the purpose and policies that are used for different grade levels. Manuscript 3 examines programs such as participation in a Research Practice Partnership. Lastly, *Practices* are the processes of teaching which include the interactions between teachers and students that include the purpose, policies, and programs of the reform. Manuscript 4 focuses on practices by describing what CT skills teachers are engaging with in their teaching and learning, while also including descriptions of lessons that have opportunities for CT integration. This 4P model encompasses all of these research studies with the main focus of this research concentrating on examining the purpose of CT skill integration in elementary science classrooms, examining teacher practices and district programs, and providing recommendations on how to develop appropriate policies.

The following research questions guiding these studies are:

- 1) What is the definition of computational thinking and the skills involved? -
Manuscript 1
- 2) What are the factors (teachers and district characteristics) related to the use of these CT skills? - Manuscript 2
- 3) How are elementary teachers teaching CT skills in their science classrooms? -
Manuscript 3

The following section is organized by research paper and includes the research questions, the theoretical or conceptual framework that guides each study (in addition to the 4P model), and a brief summary with the methodology used.

How is Computational Thinking defined in elementary science?

To successfully integrate CT into instruction, teachers need effective resources that support the integration of CT into their content, pedagogy, and practice. For implementation to be successful, teachers need a definition that uses precise language they can understand and use to communicate clearly to their students about what CT is. They also need precise language when developing integrated science lessons. The first paper in the manuscript aims to address the following research questions:

- How is CT defined in the literature?
- What is the process a Design-Based Implementation Research team underwent to define a teacher-friendly definition of CT for elementary teachers?

Integrating CT into the teaching of science involves understanding a body of knowledge called TPACK which identifies three types of knowledge: technology, pedagogy, content knowledge, and the interaction between and among them that educators need to understand (Koehler & Mishra, 2009). Technological Pedagogical Content Knowledge (TPACK) is an important conceptual framework to use when teaching computational thinking skills to enhance student learning experiences (Angeli, et al., 2016, p. 53). Having a curriculum is important but preparing teachers with skills and the understanding to teach it is necessary. Teachers need to understand exactly what CT is so having a clear definition is important. Once they have an understanding of what CT is they can develop lessons and use different pedagogy for integrating CT into their instruction using methods such as framing, prompting, and inviting reflection (Rich et al., 2021). In addition, they can start using different technological tools such as models and simulations (Adler & Kim, 2018; Ruberg & Owens, 2017; Sengupta et al., 2013). Obtaining new knowledge and skills can be challenging for teachers considering elementary teachers are typically generalists and have many different standards they need to teach in multiple subject areas. Having definitions with easy-to-understand language is helpful in increasing the likelihood for implementation. When implementing CT into science instruction, professional development of teachers needs to take the different components of TPACK into consideration thus providing a clear definition is essential.

In this first paper, a literature review was conducted to find out exactly what computational thinking is and how it is used in elementary science. The literature review was conducted between 2006 to 2021 to get closer to a shared definition of CT and its component parts for K–5 science education. Findings indicated there are numerous CT

definitions in the literature with many similar ideas and elements, however, there are enough differences making it difficult for teachers to integrate into their instruction. The literature review revealed there is no shared definition between the computer science and education community and that there is a need for consensus building around shared CT language.

The paper continues to address the need for a clear definition by describing the process a Design-Based Implementation Research (DBIR) team underwent to create a teacher-friendly definition for CT and its component parts. In addition to a critical review of the literature, the process of the DBIR team involved analysis of standards, value-mapping across multiple stakeholders, analysis of written responses from elementary teachers on a CT survey, and deep discussions. The data collected and analyzed from these sources was combined to develop a teacher-friendly definition of CT to support implementation for elementary teachers.

The Design-Based Implementation Research (DBIR) team was composed of 25 members, including myself, with various expertise and current job titles including elementary math and science coaches, teachers (covering all grade levels K–5), district curriculum coordinators, and university educators across STEM+C content from a northeast state. This group met to collaborate in monthly face-to-face (in person and virtual) research meetings for three years (January 2019–August 2021) to determine areas of need for the continuous improvement of STEM+C in elementary schools by integrating computational thinking.

The research from this study is helpful for curriculum designers, policy makers, and teachers by giving a common language for developing programs at the elementary

level that will lay the foundation for an essential component needed in achieving computer science for all.

Predicting Computational Thinking in Elementary Science using a Multilevel Model Approach

The second paper in this manuscript examines the frequency of CT concepts and approaches currently integrated into elementary science classrooms across a northeast state. This paper identifies whether teacher factors (grade level, teaching experience, professional development, confidence, and level of concern) or district-level factors (classification, socioeconomic status, and science curriculum) affect CT frequency levels, in aims to help with future implementation efforts. This study builds on the foundation for research and implementation of CT integration into science by addressing the following research questions:

- How often are Computational Thinking (CT) concepts and approaches currently taught in K–5 science classrooms?
- To what extent are teacher characteristics, such as grade level, teaching experience, professional development, confidence, and level of concern for implementing CT, related to the frequency of teaching computational thinking concepts and approaches in K–5 science classrooms?
- To what extent are district characteristics such as district classification, socioeconomic status, and participation in a Research Practice Partnership (RPP) using The Guiding Education in Math and Science Network (GEMS-Net) science

curriculum related to the frequency of teaching thinking concepts and approaches in K–5 science classrooms?

This paper considers Urie Bronfenbrenner’s Ecological Systems Theory.

According to Bronfenbrenner (1979), different layers or systems in a child’s environment influence their development and learning. These different environments are nested within one another in different layers that interact directly and indirectly. The first layer closest to the child is the microsystem, which is the immediate environment. This layer consists of people who directly influence the child such as teachers and parents. The mesosystem is the second layer and consists of the relationships between the different structures in the microsystem and how they affect the child. The next layer is the exosystem, which is the indirect environment that can influence the development of the child. The outermost layer is the macrosystem, which consists of social and cultural values that influence the inner-layer systems. The last layer is the chronological layer which is the changes over time which can be both external and internal changes in the child. For this research, the microsystem and exosystem were layers of interest since the microsystem consists of the qualities of the teachers and how they influence the students, and the exosystem includes the indirect environment or the district level variables that influence the child.

This second research paper contributes to the elementary CT integrated science literature in that most studies are qualitative, and this study is quantitative. This study also provides a better understanding of variables affecting CT teaching frequency. Teacher-level factors including experience, grade level, and confidence significantly influence CT in the elementary classrooms as well as the district-level factor of being part of a Research-Practice Partnership. This study reveals the benefits of belonging to a

Research Practice Partnership when implementing an innovation such as integrating CT due to the support it can provide to teachers.

This study uses a portion of quantitative survey data that is part of a larger research STEM+C study entitled *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (Sweetman, 2018–2020) (1813224). The survey used was taken by elementary school teachers who teach science in grades K–5. The data includes teacher responses to questions asked in a cross-sectional web-based survey that took teachers approximately 30 minutes to complete concerning their current practices in the classroom. Teacher-level and district-level demographics and opinions were collected. The survey was educative in describing the concepts (decomposition, abstraction, pattern recognition, algorithm) and approaches (perseverance, creativity, tinkering, debugging, collaboration) involved in CT by using definitions, pictures, and examples of the different elements modified from Barefoot Computing at School curriculum. Frequency levels of use for each concept and approach integrated in the science lesson were asked in the survey. Data was analyzed using descriptive statistics for frequencies of the CT concepts and approaches. Hierarchical Linear Modeling (HLM) was used to analyze teachers (level-1) nested within districts (level-2) to allow for less assumptions from between and within group differences (Raudenbush & Bryk, 2002).

This research brings awareness to the complex systems associated with elementary school instruction that impact the implementation of new innovations. Based on the data from teachers' surveys, it is clear that there is exciting potential to integrate

CT into daily science instruction and there are various factors that will help with implementation.

Teacher practices for CT implementation in elementary science

The purpose of the third paper is to examine the current landscape of what exists for integrated CT in science classrooms throughout a northeast state in the United States and describes science lessons that can be enhanced and extended through the analysis of written responses from a survey. This analysis aims to answer the following research questions:

- How often do K–5 teachers teach science? How long do K–5 teachers teach science lessons?
- How often and in what ways are Computational Thinking (CT) concepts and approaches currently taught in K–5 science classrooms?
- How do K–5 science teachers describe lessons where they include the different CT concepts (decomposition, abstraction, algorithms, or pattern recognition) or approaches (tinkering, debugging, creating, persevering, and collaboration) in the teaching and learning process, and how accurate are they?

A conceptual framework used for integrating CT into curriculum and that guides this research is the Exist, Enhance, Extend Framework (Waterman et al., 2020). There are three levels in this framework that increase in complexity when implementing CT. The first level referred to as “Exist” is when teachers can recognize different CT concepts and approaches that are present in the curriculum and make connections to how they relate to technology and computers. “Enhance” is when additional tasks or lessons are created that

are not central to the lesson but make clear connections to computing concepts. The last level is “Extend,” which is when new lessons or sequences are created that promote CS exploration through activities like programming. For this research, we are focusing on the “Exist” part of this framework.

The third paper explores the current CT practices of K–5 science educators using survey data collected from teachers. Analyzing the survey data uncovers how much teaching time is spent teaching science and the percentage of lessons that have CT concepts and approaches present in the science instruction. In addition, the survey data also describes examples of what these lessons look like. The research reveals that teaching CT concepts and approaches are present in curriculum all throughout different districts and that teachers can identify opportunities for CT concepts and approaches with accuracy when given descriptions, examples, and pictures. CT can be integrated into what teachers are already doing for lessons in the classroom.

The methodology used in this study is an embedded mixed methods design with qualitative data used as supplementary to the quantitative data. This study uses both quantitative and qualitative data collected from the survey. The quantitative data analysis involves determining the average teaching time in science and the percentage of lessons that show potential for integrating the different CT concepts and approaches as reported by teachers. The qualitative data from the survey consists of responses to open-ended questions that give a deeper meaning to the information gained by the quantitative data. This data consists of teachers describing examples of the types of lessons where they integrate the CT concepts and approaches in their science instruction. Accuracy of

examples given by teachers was cross-checked through an inter-coder agreement with members of the DBIR team.

This analysis also gave information about how CT concepts and approaches are present in a substantial portion of the teaching time spent on science instruction. With CT becoming an additional skill we want our children to have and there being a limited time available for teaching science, we need to consider increasing the length of time science is being taught.

Summary of Three Paper Manuscript

The papers in this dissertation are meant to serve as publishable manuscripts, and each paper reviews various aspects of computational thinking in elementary science. All three papers are exploratory in nature and attempt to reveal the component parts of CT for a better understanding of how CT concepts and approaches integrate authentically into elementary science instruction. However, each paper employs a different analysis, while at the same time each individual manuscript is informed by, and is intended to build upon, findings from the other manuscripts.

These manuscripts contribute to the body of CT knowledge in that they provide an analysis of what is happening in elementary classrooms and what some of the needs are for implementation efforts. The first paper addresses how CT is defined in the literature and how the computer science and education communities need to come to consensus on a shared definition so implementation efforts can be more viable. The second paper determines factors such as grade level, experience, and confidence are significant predictors of the usage of CT skills. Belonging to a Research Practice Partnership is also explored in this paper. The third paper reveals the time teachers are

spending on science instruction and the frequency of using CT concepts and approaches in the classroom. This paper also provides descriptions of what teachers are doing when they see opportunities for CT concepts and approaches in their instruction and the implications for changing policy.

From this research policymakers and district leaders can be more intentional in supporting both teachers and students in developing an understanding of core computing skills needed for a solid foundation in computer science applications. This knowledge will help with guiding science teacher professional development for integrating CT into what they already do in the classroom, inform curriculum developers of areas of need, and educate policymakers of what baseline practices exist and factors that help with integration. This dissertation will end with a concluding chapter summarizing the findings and implications, limitations, and recommendation for future research.

CHAPTER 2

HOW IS COMPUTATIONAL THINKING DEFINED IN ELEMENTARY SCIENCE?

Currently in preparation for the *Journal of Tech Trends*

Department of Education, University of Rhode Island

Kingston, Rhode Island, 02881

Abstract

In recent years, researchers and K–12 practitioners are challenged to incorporate computer science into their daily lessons by integrating an essential component of computing identified as computational thinking. Computational thinking (CT) is a foundational skill of computer science to which all elementary students should have equitable access for learning and engaging in throughout their day (Code.org, CSTA, & ECEP Alliance, 2020). For the integration of CT into science instruction to be successful, teachers need to be able to communicate clearly what CT is and how it is defined through precise language. The purpose of this research is to explore the meaning of CT at the elementary level through a critical review of the literature, analysis of standards, value-mapping across multiple stakeholders, written responses from elementary teachers on a CT survey, and deep discussions of a Design-Based Implementation Research Team (DBIR). The data collected and analyzed from these sources was combined to develop a definition of CT that would support implementation for elementary teachers. In addition to supporting elementary teachers in understanding CT, the results also can inform policy makers, curriculum designers, and researchers to create programs that are specifically aimed and appropriate for elementary classrooms.

Introduction

In 2016, former President Obama, in his State of the Union Address, set the goal of “Computer science for all”, resulting in 4 billion dollars allocated in the budget to increase access and establish high-quality computer science learning opportunities for students in K–12 classrooms (CS for All, 2021). The focus of this goal was to address the needs of the workforce and allow underrepresented groups more opportunities to participate in computer science. This announcement led to many states aiming to bring high-quality learning experiences in computer science to all their students. The National Science Foundation began funding research and development for bringing computer science to all schools. Educators and researchers began studying and implementing goals of preparing K–12 teachers for integrating computer science into daily lessons and providing instructional materials. When considering how to integrate computer science, they focused on an essential component of computing termed “computational thinking” (Code.org, CSTA, & ECEP Alliance, 2020).

Computational thinking (CT) is identified as a foundational skill children need to become problem solvers, innovators, and informed citizens in today’s technological world (Barr and Stephenson 2011; Grover and Pea 2013). Grover (2018) contends that CT is best learned when it is embedded in class subjects, taught in context using an interdisciplinary approach, and begun in the early years of schooling. CT is believed to be at the core of all STEM disciplines (Henderson, Cortina, Hazzan, & Wing, 2007). For the integration of CT into subject areas to be effective in K–5 education, teachers need to build awareness of what CT is and how it is defined through precise language so they can communicate ideas clearly with their students.

However, there is no agreed-upon definition used for integrating CT into K–12 education that applies across all content and grade levels (Gane et al., 2021). The variety of definitions contain similar language but consist of different elements that make up CT. Many of the definitions refer to CT as a process to solve problems. However, problems at the elementary level look quite different when compared to problems at the high-school level. The discipline in which CT is taught also varies and can be domain specific (Denning & Tedre, 2019). The computer science and education communities need to align their terminology associated with computational thinking and develop consensus about the skills involved so teachers will be able to synthesize research and programs and put innovative ideas into practice.

The research presented in this paper narrows the field of CT to include a focus on CT in elementary settings and CT that is integrated into science instruction. Children who receive CT instruction in early education can develop computer science foundational skills which may help develop positive attitudes towards STEM disciplines (Karpinski et al., 2021; Maltese & Tai, 2010). The importance of CT in early education is clear, yet implementation of CT at the elementary level remains low since the literature is at an early stage of development for knowing how to teach this skill (Kalelioglu, et al., 2016). A clear definition for teachers of our youngest learners is necessary to support implementation in elementary schools and to build upon when developing learning pathways for subsequent grade levels. In addition to looking at CT integration in early education, this research focuses on the integration of CT in science instruction. The rationale is that computing concepts and practices have become a fundamental part of the work of professional scientists (Denning & Tedre, 2019). McGinnis et al. (2019) believe

that opportunities for CT in science for young learners helps make science more mentally challenging, interesting, and fun because it allows for children to think about science in a different way.

The Next Generation Science Standards (NGSS) are the national standards to guide science instruction across the K–12 system. The NGSS identify computational thinking as one of the eight science and engineering practices that all students should be learning and doing throughout their education (NGSS, 2013). When CT is integrated into a core subject area that is required for all students, then it is more likely to reach all children, allowing for a more diverse population to develop an interest in and pursue STEM and computing careers. Also, defining CT in a disciplinary context can help contribute to teachers' abilities to understand and use CT in their practice (Breslyn & McGinnis, 2019).

There are several barriers to the implementation of integrating CT into elementary science instruction: many teachers are inexperienced with teaching CT; there is limited time for planning and developing curriculum that integrates CT; and other school initiatives take priority (Google & Gallup, 2016; Ketelhut et al., 2020). Regardless of these obstacles, elementary school leaders are still tasked with including computer science in their daily instruction. To successfully implement an integrated approach, educational leaders need evidence-based outcomes to make decisions on curriculum, and teachers need effective resources that support the integration of CT into their content, pedagogy, and practice. For implementation to be successful, teachers need a definition that uses precise language they can use to communicate clearly to their students about what CT is and to use when developing integrated science lessons. This paper aims to

share the process a Design-Based Implementation Research (DBIR) team underwent over a three-year period (January 2019–August 2021) to develop a teacher-friendly definition of CT by examining the literature, cross-walking standards, value-mapping across multiple stakeholders, analyzing written responses from elementary teachers on a CT survey, and having numerous discussions as a DBIR Team.

Conceptual Framework of TPACK

Technological Pedagogical Content Knowledge (TPACK) involves understanding types of knowledge: technology, pedagogy, content knowledge, and the interaction between and among them (Koehler & Mishra, 2009). It is an important conceptual framework to use when integrating CT into curriculum (Angeli et al., 2016; Koehler & Mishra, 2009). Having knowledge of science content, pedagogy, technology, and an understanding of how CT and the elements that comprise it are defined is essential for integrating CT into discipline specific instruction. Kang et al. (2018) conducted a survey where teachers ranked their pedagogical content knowledge and confidence in teaching the engineering practices of the Next Generation Science Standards and found that using mathematics and computational thinking scored the lowest out of the eight engineering practices. Despite it being ranked low, integrating computational thinking practices into science curricula is beneficial for student engagement and deeper learning of science (Cateté et al., 2018). Practitioners need to be informed about how to best place CT in the curriculum while considering factors such as grade levels, contexts, and what is developmentally appropriate (Bocconi et al., 2016). TPACK gives emphasis to understanding a body of knowledge teachers must possess for successful integration of CT.

Computational Thinking Background

Computational thinking was first introduced into the literature by Seymour Papert in 1996 but gained recognition in 2006 when Jeanette Wing advocated for all K–12 students to engage in CT, defining it as “solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” (Wing, 2006, p. 33). Wing’s definition was further refined in 2011 as “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” (Cuny, Snyder & Wing, 2010, cited in Wing, 2011, p. 20).

Although Wing’s definitions are often referenced, researchers in the computer science and education fields added to her definition and have defined CT in numerous ways. Barr and Stephenson (2011) defined computational thinking as “an approach to solving problems in a way that can be implemented with a computer” (p. 51). They provided examples of how CT can be easily integrated into the subject areas of math, science, social studies, and language arts in K–12 grades. The International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) (2011) gathered feedback from close to seven hundred computer science teachers, researchers, and practitioners to create an operational definition of computational thinking as:

formulating problems in a way that enables us to use a computer and other tools to help solve them, logically organizing and analyzing data, representing data through abstractions, such as models and simulations, automating solutions through algorithmic thinking, identifying, analyzing, and implementing possible

solutions with the goal of achieving the most efficient and effective combination of steps and resources, and generalizing and transferring this problem-solving process to a wide variety of problems. (p. 13)

In addition, ISTE and CSTA (2011) also list different dispositions for engaging in CT which include confidence in dealing with complexity, persistence in working with difficult problems, tolerance for ambiguity, the ability to deal with open-ended problems, and the ability to communicate and work with others to achieve a common goal or solution. Although these attributes are important for CT, it makes it less clear what is most essential for CT when putting it into practice in the classroom.

Aho (2012) defined CT as “the thought processes involved in formulating problems so their solutions can be represented as computational steps and algorithms” (p. 832). He stressed the importance of finding appropriate models when producing solutions to problems and the need for inventing new models based on the increase of accessibility to complex data. Brennan and Resnick (2012) added new ideas to prior definitions by describing CT as being composed of three dimensions: computational concepts, practices, and perspectives. They categorized concepts as sequences, loops, events, parallelism, conditionals, operators, and data and used the Scratch program for defining these elements. These concepts are important elements in computer programming. They described the CT practices as being incremental and iterative, testing and debugging, reusing, and remixing, and abstracting and modularizing. The perspectives include expressing, connecting, and questioning. Although there is overlap with some of the component parts listed in prior definitions, such as abstraction and algorithms, some new

terminology is also introduced along with the idea of whether computer programming should be part of the CT definition.

In addition to CT being defined in the computer science and education fields, CT was introduced nationally into the science discipline in 2013, when it was listed in the Next Generation Science Standards (NGSS) as one of the eight engineering practices (NGSS Lead States, 2013). The NGSS states computational thinking as “strategies for organizing and searching data, creating sequences of steps called algorithms, and using and developing new simulations of natural and designed systems” (NGSS, Appendix F, p. 10). This definition does not include computer programming. However, the NGSS are categorized into different grade spans which vary in the use of CT used in the performance standards. For example, there are thirteen performance standards that include CT at the high-school level, whereas there are only two at the middle and elementary levels.

Grover and Pea (2013) explain CT is a problem-solving process with the following characteristics: (1) formulating problems in a way that enables us to use a computer and other tools to help solve them; (2) logically organizing and analyzing data; (3) representing data through abstractions such as models and simulations; (4) automating solutions through algorithmic thinking; (5) identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources; and (6) generalizing and transferring this problem-solving process to a wide variety of problems. Most of the definitions leading up to this point involve the process of problem-solving or thought processes that are used to conduct a solution

whether it be by a computer or a human being in a systematic, efficient, and effective way.

The United Kingdom program, Computing at School, indicates CT is about looking at a problem in such a way that a computer can help solve the problem involving a two-step process: (1) thinking about the steps needed to solve a problem, and (2) using technical skills to get the computer working on the problem. CT is broken down into six concepts: logic, algorithms, decomposition, patterns, abstraction, and evaluation. The five approaches are tinkering, debugging, persevering, creating, and collaboration (Barefoot, 2014). Yadav et al. (2014) offer similar but fewer components when breaking down CT; their components are problem decomposition, abstraction, logical thinking, creating algorithms, and debugging. Voogt et al. (2015) analyzed many of the definitions and argued that including a broad list of general terms makes CT indistinct from other 21st century skills. They believe we should not try to give a definition with necessary conditions but rather with possible conditions based on looking at similarities and relationships concerning CT in discussions. They believe it will lead to a more concise description for integrating CT and what matters most about CT. However, this provides little guidance for practicing teachers who are not well-versed in CT discourse on how to even begin implementation while juggling many other priorities on their plates. In addition, the definitions are not all explicit about whether a computer needs to be used in the process.

Weintrop et al. (2016) created a taxonomy for integrating CT into science and math instruction after analyzing high-school STEM classroom activities. The taxonomy contains four CT categories: data practices, modeling and simulation, computational

problems solving, and systems thinking with 22 CT practices found within the four categories. CT is also found all throughout the K–12 Computer Science Framework (2016), which introduces the foundation of computer science to all students, including at the elementary level. This framework is classified into five computational concepts and seven practices. The concepts are computing systems, networks and internet, data and analysis, algorithms, and programming, and impacts of computing. The practices include collaborating around computing, communicating about computing, creating computational artifacts, developing, and using abstractions, fostering an inclusive computing culture, recognizing, and defining computational problems, and testing and refining computational artifacts. These frameworks can be useful in organizing the ideas of CT into categories, but they also add another layer of complexity to the understanding of CT by elementary teachers.

Some of the more recent definitions are still not in agreement of whether a computer is needed or not. Shute et al. (2017) defined CT as “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). This conceptual foundation is categorized into six main facets: decomposition, abstraction, algorithm design, debugging, iteration, and generalization. This definition made it clear that a computer was not necessary for the CT process. However, one year later Marina Bers (2018) described CT as a systematic way of thinking that encompasses analytic and problem-solving skills as well as language for expression and communication. Bers posits that coding or programming needs to be part of computational thinking when integrated at the elementary level. She explains it is a

new literacy needed for the 21st century; it is a new way of thinking, communicating, and expressing ideas like how writing is just as important as reading. She adds that those who cannot read or write are left out of the power structures, and their thoughts are not heard; likewise, students need to be able to code to be producers and not only consumers of digital products. Bocconi et al. (2018) contends that programming can make CT concepts concrete which leads to powerful ideas. On the other hand, Grover, and Pea (2018) argue “Although programming is a key vehicle to teach and learn CT, it can be taught in the classroom with or without a computer or programming” (p.35). Martínez-García (2021) contends that understanding CT is different from programming. It is a way of thinking that is based on abstraction and critical thinking.

The CT definitions all listed have many similar ideas and elements which include having skills, habits, and attitudes to solve complex problems. However, there are enough differences making it difficult for teachers to integrate into their instruction. Terminology of the different elements also varies and is often referred to as concepts, practice, or perspectives along with different definitions for these elements. When implementing into grade levels, educators need more information on what is developmentally appropriate and whether specific elements that make up CT are discipline specific and whether or not they are necessary.

Literature Review of CT in K–5 Elementary Science

A definition for CT for K–5 science education is needed for successful integration. Teachers need to be able to communicate clearly what CT is and how it is defined through precise language. A literature review was conducted of studies from 2006 to 2021 to get closer to a shared definition of CT for K–5 science education. To

identify relevant studies, an electronic search was conducted using the databases (EBSCOhost) Academic Search Complete, Education Full Text (H.W. Wilson), ERIC (EBSCOHOST), and SCOPUS. The search looked for articles that focus on integrating CT in elementary science classrooms and was limited from January 2006 to December 2021 to start with the year Jeanette Wing brought CT to the public's attention and to include current literature. The keywords "computational thinking," "science," and "elementary" or "primary" were used to conduct the search. The studies were included in the review if they met the following criteria: (a) make an explicit reference to the term "computational thinking" in the title, abstract, or keywords; (b) be written in English language; (c) have a typical form of a scientific paper (i.e., no posters, roundtables, work-in-progress papers, short papers, etc.); and (d) focus on CT and elementary-level science education. A total of 53 articles met the initial search criteria, but after a closer look at each paper's title, abstract, and content, 33 papers were excluded for not containing all relevant inclusion criteria. The final review consisted of 20 papers (Appendix A) with 17 different first authors.

These 20 research studies (Table 1) showed a great deal of variance in defining CT, with 11 different main definitions and with 5 of the authors clearly stating there is no agreed-upon definition for CT while settling on a few main definitions. Many of the articles explicitly use Jeanette Wing's initial definition. In fact, 16 out of 20 of the articles reference her definition of CT, and 9 out of 20 of the articles use it as one of their main definitions. The next most common definition, used by five of the research articles, is the definition from the Next Generation Science Standards (NGSS, 2013). This is followed by two pairs of research articles sharing the same definition (Barr &

Stephenson, 2011; Yadav et al., 2014, 2018). The remaining articles provide different definitions used by only their research study for elementary science (Berland & Wilensky, 2015; Brennan & Resnick, 2012; Eshan, 2021; ISTE & CSTA, 2011; Massachusetts Digital Literacy and Computer Science Curriculum, 2016; Rich, 2019; Shute et al., 2017). This variance clearly shows that even when CT is defined with a particular subject and educational level, there still is not a clear-cut definition for teachers to use when integrating CT into their daily teaching.

Weintrop et al.'s (2016) taxonomy is commonly referenced in this literature review. Their taxonomy embeds CT into science instruction using four categories: data practices, models, and simulations, computational problem-solving, and systems thinking is commonly referenced in this literature review. This framework was specifically referenced by six of the research articles reviewed with twelve of the articles using components of the taxonomy, mainly data practices. Breslyn and McGinnis (2019) chose to adopt Weintrop's framework for their research study and questioned the developmental appropriateness of systems thinking for elementary science since the taxonomy was created by analyzing 24 high-school science lessons. They found that systems thinking was appropriate to use at the elementary level if age-appropriate examples of CT and relevant science content were provided. Also, limiting the depth and complexity of the CT practices is also necessary since there is a different range of abilities at the elementary level.

Most of the research studies reference using data but employ various terminology such as pattern recognition, data collection, data analysis, or data representation. Weintrop et al. (2016) broke data practices down into collecting, creating, manipulating,

analyzing, and visualizing data. Analyzing and interpreting data are essential components of scientific inquiry and are part of the NGSS science and engineering practices like the data practices outlined in the CT taxonomy. Modern technology allows for opportunities to use enormous amounts of data, so being able to understand how to organize data in a form that can show patterns and relationships that can be communicated is critical for learning both computer science and general science along with making new scientific discoveries (Martínez-García, 2021; National Research Council, 2012).

Table 1*Research Studies on CT Integrated in Elementary Science*

Authors	Title	Year	Main Definition (Many authors explicitly stated there is no consensus definition)	<i>Essential knowledge of computational thinking</i> Main concepts/ competencies/ aspects	<i>Key actions, processes, and proficiencies</i> Main practices/ approaches/ dispositions listed	Grade level the research was conducted in
Breslyn & McGinnis	Investigating preservice elementary science teachers' understanding of climate change from a computational thinking systems perspective	2019	Berland & Wilensky (2015) “the ability to think with the computer-as-tool”	data collection, data analysis, data representation, problem decomposition, abstraction, algorithm & procedures, automation, parallelization, and simulation	data, modeling and simulation, computational problems solving, and systems thinking (Weintrop, 2016)	pre-service teachers
Dickes et al.	Sociomathematical norms for integrating coding and modeling with elementary science: A dialogical approach	2020	(Wing, 2006)—an analytic problem solving and design approach fundamental to computing—with K–12 STEM classrooms	problem representation, abstraction, decomposition, simulation, verification, and prediction	modeling and simulation	Third grade 15 students and 1 teacher – part of RPP

Ehsan et al.	Computational thinking embedded in engineering design: Capturing computational thinking of children in an informal engineering design activity	2021	This research group utilized the computational thinking framework developed by their own research team after compiling, summarizing, and synthesizing several computational thinking frameworks. Only defined CT competencies	abstraction, algorithms, and procedures, troubleshooting and debugging, pattern recognition, problem decomposition, simulations Defined CT competencies. <i><u>Abstraction</u>-Identifying and utilizing the structure of concepts/main ideas</i> <i><u>Algorithms and procedures</u>-Following, identifying, using, and creating an ordered set of instructions</i> <i><u>Troubleshooting and debugging</u>- Identifying and addressing problems that inhibit progress toward task completion</i> <i><u>Pattern recognition</u>-Observing patterns, trends, and regularities in data</i> <i><u>Problem decomposition</u> – Breaking down data, processes, or problems into smaller and more manageable components to solve a problem</i> <i><u>Simulations</u> Developing any type of model or a representation</i>	engineering design problem scoping idea generation idea representation design evaluation	10 K–2 nd grade students
Ehsan et	Computer science	2019	“Solving problems,	abstraction, algorithms and	analyzing and	2nd grade

al.	unplugged		designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” (Wing, 2006, p. 33).	procedures, automation, data collection, data analysis, data representation, debugging /troubleshooting, decomposition, parallelization, simulations, pattern recognition	interpreting data, developing and using models	class field trip
Hestness et al.	Professional knowledge building within an elementary teacher professional development experience on computational thinking in science education	2018	Definition for Elementary level- CT is an approach to solving problems in a way that can be implemented with a computer. Students become not merely tool users but tool builders. They use a set of concepts, such as abstraction, recursion, and iteration, to process and analyze data, and to create real and virtual artifacts. CT is a problem-solving methodology that can be automated and transferred and applied across subjects. (Barr & Stephenson, 2011) Definition for science “Strategies for organizing and searching data, creating sequences of steps called algorithms, and using and developing new simulations of natural and designed systems” (NGSS, Appendix F, p. 10)	data collection, data analysis, data representation, problem decomposition abstraction, algorithms & procedures automation, parallelization, simulation	data, modeling and simulation, computational problems solving, and systems thinking (Weintrop, 2016) -Confidence in dealing with complexity, -Persistence in working with difficult problems, -The ability to manage ambiguity, -The ability to deal with open-ended problems, -Setting aside differences to work with others to achieve a common goal or solution, and - Knowing one's strengths and	13 mentor teachers 3rd and 5th grade

					weaknesses when working with others.	
Israel et al.	Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis	2015	Formulating problems in a way that enables us to use a computer and other tools to help solve them; logically organizing and analyzing data; representing data through abstractions such as models and simulations; automating solutions through algorithmic thinking; identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combinations of steps and resources; and generalizing and transferring this problem solving process to a wide variety of problems (ISTE & CSTA, 2011) and (Wing, 2006, p. 33).	abstractions, data analysis, automation, algorithmic thinking, generalization	data modeling and simulation problem solving	2nd grade to 5th grade 7 teachers and 2 administrators continued...
Jaipal-Jamani & Angeli	Effect of robotics on elementary preservice teachers' self-efficacy, science learning,	2017	CT involves engaging in processes such as problem decomposition, abstraction, logical thinking, creating algorithms, and debugging	algorithms, abstraction, debugging, programming	data analysis	21 elementary pre-service teachers grades 4–8

	and computational thinking		(Yadav et al., 2014) and (Wing, 2006).			
Kang et al.	Exploring elementary teachers' pedagogical content knowledge and confidence in implementing the NGSS science and engineering practices.	2018	“Strategies for organizing and searching data, creating sequences of steps called algorithms, and using and developing new simulations of natural and designed systems” (NGSS, Appendix F, page 10)		science and engineering practices	17 2nd grade teachers
Kaya et al.	Examining the impact of a computational thinking intervention on pre-service elementary science teachers' computational thinking teaching efficacy beliefs, interest, and confidence	2019	(Wing, 2006) and (NGSS, Appendix F, p. 10)		science and engineering practices	56 pre-service elementary teachers
Kaya et al.	Measuring computational thinking teaching	2020	(Wing, 2006) and (NGSS, Appendix F, p. 10)	decomposition, algorithms, abstraction, parallelization <i><u>Decomposition- Solving</u></i>	science and engineering practices	35 pre-service elementary

	efficacy beliefs of preservice elementary teachers			<p><i>complex problems by breaking it into smaller, manageable parts.</i></p> <p><u>Algorithms</u>-systematic rules (step-by-step) in problem solving that are relevant in our daily life</p> <p><u>Abstraction</u>- Applying generalizations that can be transferred to new problem solutions.</p> <p><u>Parallelization</u> -Conducting two independent events simultaneously</p> <p><u>Debugging</u> -Finding issues in a solution or program, then fixing and modifying the solution to improve the original outcome.</p>		teachers
Ketelhut et al.	Teacher change following a professional development experience in integrating computational thinking into elementary science	2020	(Wing, 2006, p. 33).	problem-solving, decomposition, pattern recognition, abstraction, algorithm, evaluation	data, modeling and simulations, computational problem-solving practices, and systems thinking. (Weintrop, 2016)	13 mentor teachers 3rd–5th grade
Luo et al.	Exploring the	2020	Computational thinking	abstraction, decomposition	iteration,	2 elementary

	evolution of two girls' conceptions and practices in computational thinking in science		involves three key dimensions: computational concepts, computational practices, and computational perspectives. (Brennan & Resnick, 2012, p. 3)	sequences, loops, events, parallelism, conditionals, operators, and data	debugging and abstraction: and the perspectives are expressing, connecting, and questioning in programming.	girls 3rd grade - Asian American 5th grade - Caucasian
McGinnis et al.	Preservice science teachers' beliefs about computational thinking following a curricular module within an elementary science methods course	2020	(NGSS, Appendix F, p. 10) and (Barr & Stephenson, 2011)		data practices, models and simulations, computational problem-solving, systems thinking. (Weintrop, 2016)	39 1st–5th grade pre-service teachers
McGinnis et al.	Preservice science teachers' intentions and avoidances to integrate computational thinking into their science lesson plans for young learners	2019	(Wing, 2006, p. 33)		data practices, models and simulations, computational problem-solving, systems thinking. (Weintrop, 2016)	39 pre-service teachers

Moore et al.	Multiple representations in computational thinking tasks: A clinical study of second-grade students	2020	“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” (Shute, 2017, p. 151)	Decomposition, abstraction, algorithm, debugging, iteration, generalization	representations	3 2nd-grade students
Ogegbo & Ramnarain	A systematic review of computational thinking in science classrooms	2021	The ways of thinking, or mental habits that computer scientists use and that such mental habits might be useful for science and/or math’s inquiry. Yadav et al. (2018) (latest definition)	decomposition, recognition of patterns, abstraction, algorithm design, automation <i>Decomposition- breaking down a complex task into smaller, and more manageable components</i> <i>Recognition of patterns- identifying and defining trends or patterns within a problem</i> <i>Abstraction- identification of particular similarities and differences between comparable problems to work towards a solution</i> <i>Algorithm design-the development of step-by-step guidelines for solving a problem and can be used again to answer similar problems</i> <i>Automation- the use of</i>	data practices, models and simulations, computational problem-solving, systems thinking. (Weintrop, 2016)	23 Articles integrating CT into science 3 elementary level articles continued....

				<i>technological tools to mechanize problem solutions</i> (ISTE & CSTA, 2011; Kalelioglu et al., 2016; Yadav et al., 2018).		
Rich et al.	Teacher implementation profiles for integrating computational thinking into elementary mathematics and science instruction	2020	(Wing, 2006)	abstraction, decomposition, debugging, and patterns. Own definitions <u>Abstraction</u> - <i>Reducing complexity by focusing on important elements of a problem or situation</i> <u>Decomposition</u> - <i>Breaking apart a complex problem or situation to make it more manageable</i> <u>Debugging</u> - <i>Systematically finding and correcting problems and errors</i> <u>Patterns</u> - <i>Looking for similarities between new problems and problems that have already been solved</i>	framing, prompting, and inviting reflection.	8 elementary teachers 2nd, 3rd, 4th, 5th grade
Rich et al.	Computational thinking, mathematics, and science: Elementary teachers'	2019	"CT involves processes central to computer science, such as creating abstractions based on only the most relevant information in a problem, developing	Abstraction, algorithms, decomposition, automation, generalization, debugging Definitions for parts <u>Abstraction</u> - <i>the process of making an artifact more</i>	make sense of problems and persevere in solving them. developing and using models.	12 teachers 1st, 3rd, 4th, 5th grade

	perspectives on integration		algorithms to accomplish specific tasks, and decomposing problems into smaller and more tractable pieces” (Rich et al., 2019, p. 167)	<p><i>understandable through reducing the unnecessary detail.</i></p> <p><u>Algorithmic thinking</u> - a way of getting to a solution through a clear definition of the steps.</p> <p><u>Automation</u> - a labor saving process in which a computer is instructed to execute a set of repetitive tasks quickly and efficiently compared to the processing power of a human.</p> <p><u>Decomposition</u> - a way of thinking about artifacts in terms of their component parts.</p> <p><u>Debugging</u> - the systematic application of analysis and evaluation using skills such as testing, tracing, and logical thinking to predict and verify outcomes.</p> <p><u>Generalization</u> - associated with identifying patterns, similarities, and connections, and exploiting those features.</p> <p>(Bocconi et al., 2016)</p>		continued...
Waterman et al.	Integrating computational thinking into elementary	2020	“A problem-solving process that requires people to think in new ways to enable	abstraction, algorithms, data, modeling, and simulation (programming and	representing and analyzing data modeling using complex	3rd grade curriculum development

	<p>science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning</p>		<p>effective use of computing to solve problems and create solutions. The capacity of computers to rapidly and precisely execute programs makes new ways of designing, creating, and problem solving possible. Computational thinking is characterized by:</p> <ul style="list-style-type: none"> -analyzing, modeling, and abstracting ideas and problems so people and computers can work with them. - designing solutions and algorithms to manipulate these abstract representations (including data structures); and -identifying and executing solutions (e.g., via programming). <p>(Massachusetts Digital literacy and Computer Science Curriculum, 2016)</p>	development)	models	
--	---	--	--	--------------	--------	--

Yadav et al.	Computational thinking in elementary classrooms: measuring teacher understanding of computational ideas for teaching science.	2018	(Wing, 2006, p. 33).	logic, data collection /analysis / representation, algorithms, debugging, plugged vs. unplugged activities	constructing simulations; statistically analyzing data; and recognizing, expressing, and applying quantitative relationships	9 elementary teachers 7 elementary teachers - science Grades 2–5
--------------	---	------	----------------------	--	--	---

From the 20 research articles reviewed, 30 different components were identified making up CT. Figure 1 shows the frequency of components that were identified in two or more articles. From this analysis, it appears that *abstraction* is used the most, followed by *algorithms* and *decomposition*. When reviewing the definitions of the component parts, there are different definitions, including some original definitions created in a few of the research studies. For example, the element of abstraction was defined as identifying and utilizing the structure of concepts/main ideas (Ehsan et al., 2021); applying generalizations that can be transferred to new problem solutions (Kaya et al., 2020); identification of similarities and differences between comparable problems to work towards a solution (Ogegbo et al., 2021); reducing complexity by focusing on important elements of a problem or situation (Rich et al., 2020). Not only is there

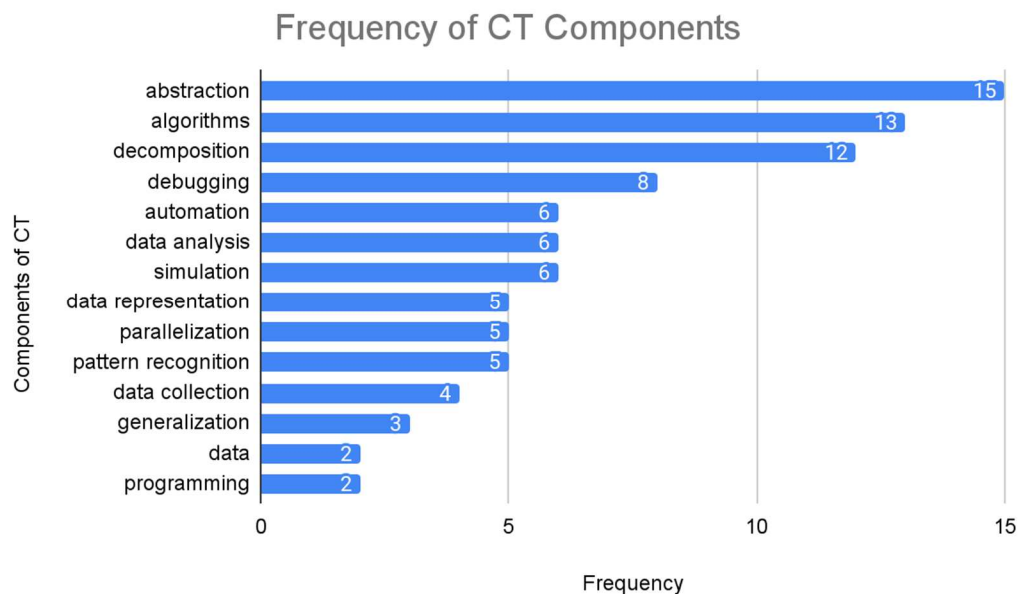


Figure 1

Frequency of CT Components Used in Research Studies

variation in the main definition of CT, but variations exist also in the components, making it difficult for implementation purposes.

Results from the research studies that address the integration of CT into K–5 elementary can be found in Figure 2. Sixty-seven percent of the research studies focused on how professional development through workshops, training, and methods courses affects in-service and preservice teachers’ knowledge and attitudes towards integrating CT into their instruction. Nineteen percent of the studies are about the engagement of students using CT skills in integrated instruction. However, the number of students reported in each study are too few for generalizations (Table 1). The remaining research studies are about designing curriculum using the exist, enhance, and extend framework, CT programming and modeling, and a literature review.

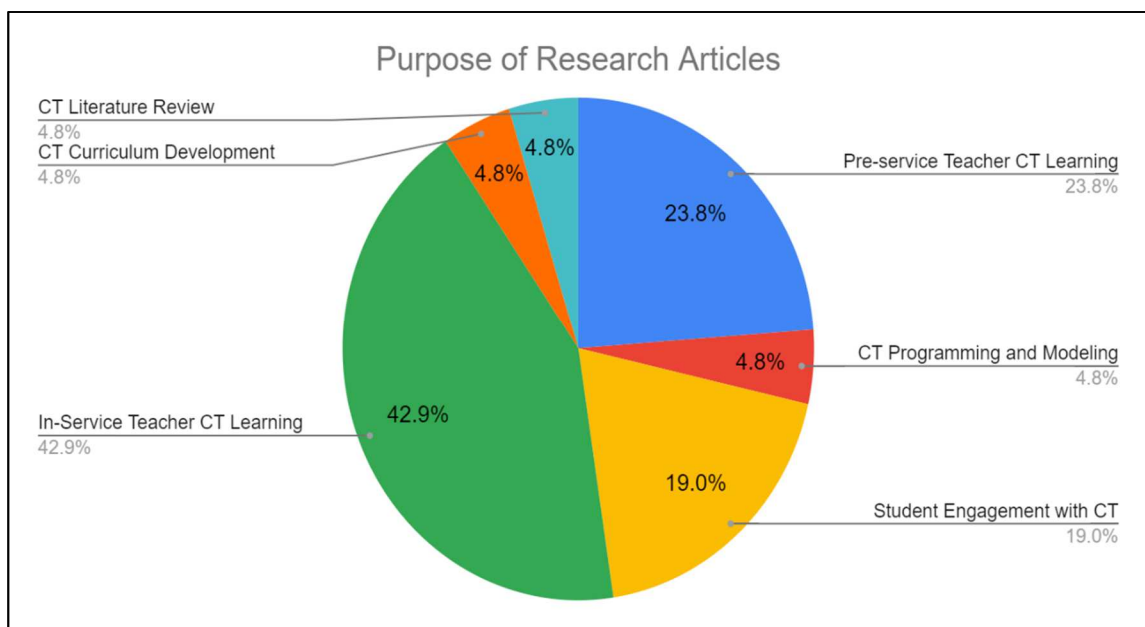


Figure 2

CT Research Studies Conducted in K–5 Elementary Science

Research methods were analyzed and 85% of the articles used a qualitative research design (Creswell, 2014). Qualitative data can provide rich insight into human behaviors and is useful for uncovering emic views by understanding the meaning and purposes attached to human activities (Guba & Lincoln, 1994) Patton (2015) contends that a small homogeneous sample is most appropriate when in-depth information is needed about a group. However, there are few quantitative studies for generalizing for integrating CT into elementary science instruction present in the literature at this time. One limitation of this literature review is that it was constrained to four educational search databases. The collection was limited to peer-reviewed journals that focused on CT in elementary science classrooms from 2006–2021 and did not include papers that may significantly contribute to our understanding of CT such as conference papers, books, or white papers. An interesting finding is that although the literature search included sources published starting in 2006, references to CT integration into elementary science did not show up until 2015, with one article. This was followed by no articles referring to integration in elementary science in 2016, and one article in 2017. In 2018, there are three articles, followed by five articles for 2019. In 2020/2021, there were ten articles published that met the search criteria, despite COVID-19 affecting research progress.

Most of the articles included in this literature review gave insight to the current landscape of research on CT in elementary instruction. This literature review added to the findings of Kalelioglu et al. in 2016 that research on CT at the elementary level is in a nascent stage. The review also highlights the need for a shared definition to support teachers with implementation of CT. The remainder of this research paper describes the

process a research team took to develop a definition for CT that teachers can use to communicate clearly to their students.

Procedure

The research performed in this segment of the paper involves analyzing artifacts produced by a Design-Based Implementation Research (DBIR) team that was part of a larger research STEM+C study entitled *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (Sweetman, 2018–2021) (1813224). The DBIR team consisted of 25 participants with various expertise and current job titles including elementary math and science coaches, teachers (covering all grade levels K–5), district curriculum coordinators, and university educators across STEM+C content. This group met to collaborate in monthly face-to-face (in-person and virtual) research meetings for three years (January 2019–August 2021) to determine areas of need for the continuous improvement of STEM+C in elementary schools in a northeast state. Throughout the different meetings, the DBIR team designed, implemented, and analyzed qualitative and quantitative survey data collected from elementary teachers across the northeast state concerning their CT practices in classroom instruction. In addition, the DBIR group investigated how to best define computational thinking through performing crosswalks of computer standards, engaging in value mapping, and numerous discussions which led to the creation of a teacher-friendly definition of CT. This paper will provide thick descriptions and analysis of the artifacts that led to the teacher-friendly definition of CT.

Crosswalk of the standards

In the beginning of the research study, the DBIR team looked critically at different computer science standards to help define computational thinking for designing the CT survey. The standards viewed were the state level Rhode Island K–12 Computer Science Education Standards (2018), the Computer Science Teacher Association (CSTA) standards (2020), the International Society for Technology in Education (ISTE) standards (2018), and the Next Generation Science Standards (NGSS) (2013).

After performing a crosswalk of the different standards and frameworks, the DBIR team discovered major discrepancies in how CT was defined and described. While there was some shared terminology, there were many differences. For instance, all the standards listed *algorithms* and *solving problems* as main parts of CT. They all listed *abstraction* too, except the NGSS. *Decomposition* was listed for CSTA and ISTE standards, but the word *modularity* was used instead for the Rhode Island K–12 Computer Science Education Standards, and *decomposition* was not listed in the NGSS. *Using data* was listed by each type of standard, but each standard used different terminology, such as “finding patterns for NGSS,” “data structure,” and “data type” for the Rhode Island Computer Science Education Standards, “analyzing data” for ISTE, and “data and analysis” for CSTA. However, terms like *parallelization*, *evaluation*, *logic*, *simulation*, *automation*, *data collection*, and *variables* were presented in only a few standards and not others.

Members of the team questioned whether programming needed to be included when teaching computational thinking, especially since the state-level standards have CT grouped together with computer programming. However, the team decided over time

programming did not need to be taught, based on support from the literature that stated solutions can be conducted with or without a computer (Grover & Pea, 2018; Martínez-García, 2021). From this crosswalk, the DBIR team realized how important it is to have a mutually shared language between the computer science community and educators.

When the DBIR team set out to define components of CT for elementary teachers, they found the Computing at Schools (CAS) framework (Barefoot Computing, n.d.) was a good starting point because it not only used accessible language for teachers but also provided examples. The research team decided to use language like that of the CAS framework because the team felt the CAS language was easy to understand and provided clear guidance for teachers since many teachers are not aware of what CT is or any of its component parts. The team also felt elementary teachers have many responsibilities learning multiple standards in different subject areas, so finding definitions with easy-to-understand language would help in increasing the likelihood for implementation.

Creating a Padlet

A Padlet (Figure 3) was created for members of the team to post lesson plans, research articles, standards, visual graphics, and examples of computational thinking that occur in K–5 classrooms throughout the research study. The materials posted on the Padlet were analyzed, and the research group found the different concepts and approaches were prevalent across the different grade levels and subject areas. After analyzing the Padlet, the DBIR team created operational definitions of the concepts and approaches of CT that were originally based on Computing at Schools concepts and approaches, except *evaluation* and *logic*, since the research team felt these two concepts were embedded in

the other concepts. The Padlet analysis also generated a discussion involving the difference between problem solving and computational thinking and how the meaning changes in different contexts. This analysis led to the first iteration of teacher-friendly definitions listed in Table 2.

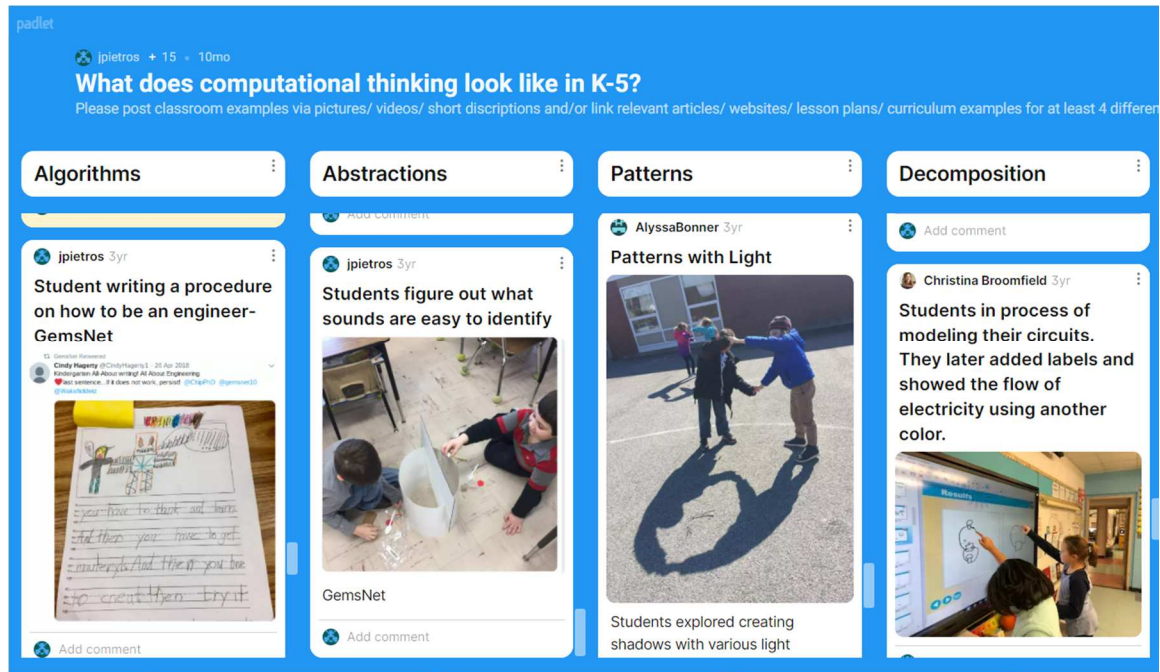


Figure 3

Padlet Created by DBIR Group

Table 2-

Teacher-Friendly Definitions for CT – First Iteration

CT Concept/Approach	Teacher-Friendly Definition
algorithm	A sequence of steps to solve a problem or achieve an outcome
abstraction	Removing unnecessary details/information with a specific focus
pattern recognition	Spotting similarities and common differences; by identifying patterns students can make predictions, create rules, solve more general problems
decomposition	Refers to breaking something up into parts. In a computational thinking context, this means breaking a complex task into simpler subtasks. In other parts of the curriculum, decomposition occurs in science as representing systems as components.
tinkering	Using manipulatives to prove a theorem or property; sticking with it until they do.
debugging	Finding mistakes and figuring out why something is not working.
creating	Using known/acquired information to design or construct and being able to share that creation with others.
perseverance	Sticking with it until the end, even if the deadline passes; intrinsic motivation
collaboration	Collaboration (as opposed to cooperation) involves mutual effort and contribution in pursuit of a shared goal. This effort and contribution do not have to be equal but are essential to achieving a creation of material or concept.

These definitions were then synthesized to make them classroom (teacher and student) friendly. They were included on a poster (Figure 4) that was printed and provided to the schools from which 560 teachers took the survey.

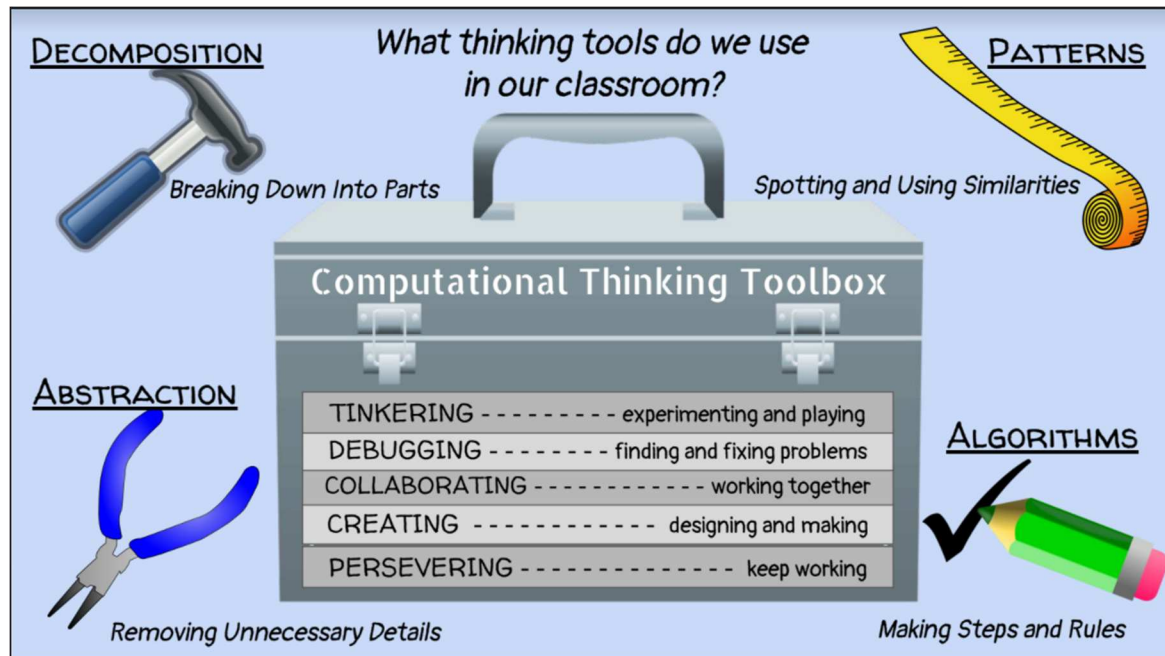


Figure 4

Poster Created with Teacher-Friendly Definitions

Analysis of survey data

Elementary school teachers in grades K–5 in a northeast state in the U.S. participated in a cross-sectional web-based survey entitled the Computational Thinking Survey as part of a larger STEM+C research study that looked at the integration of CT across multiple subject areas. Teachers responded to questions that asked them to report what types of lessons they currently taught that included CT practices. The survey was educative in describing the concepts (decomposition, abstraction, pattern recognition,

algorithm) and approaches (perseverance, creativity, tinkering, debugging, collaboration) involved in CT, using definitions, pictures, and examples of these different elements as described by the Barefoot Computing at School curriculum (Barefoot Computing, n.d.).

The responses from the survey were analyzed for accuracy and subject area and were coded by three members of the research team for interrater reliability (Tables 3 and 4). Next, the examples were coded as being “an accurate example,” “not an accurate example,” or “I don’t know.” The accuracy of the example was determined by reading the examples and using the shared definitions created by the research team from the Padlet to determine if the examples given by teachers reflected the definitions created. After all the examples were analyzed and recorded by the three members, a fourth member from the research team compared how the responses were coded and gave an overall code of accuracy based on findings being 67% or more in agreement. When there was not agreement, the response was coded as inaccurate. From this data, the subject and accuracy were quantified and analyzed.

The results indicated that teachers were able to provide accurate science examples for the concepts ranging from 63–100% accurate with an average total accuracy of 86%. They provided accurate examples of approaches ranging from 67–99% accurate with an average total accuracy of 93.5%. The team determined that the definitions could be improved to increase teacher understanding mainly for the CT concepts. Teachers struggled with providing accurate examples for *algorithms* and *abstraction*. The approaches showed better understanding by the teachers and were not as much of a focus for improving because teachers are more familiar with skills such as *collaboration* and *creating*.

Table 3

Total CT Concept Examples from CT Survey

CT Concepts	Not Accurate		Accurate		Total	
	Count	%	Count	%	Count	%
Decomposition	2	2.7	73	97.3	75	49.3
Patterns	0	0.0	22	100.0	22	14.5
Algorithms	14	36.8	24	63.2	38	25
Abstraction	5	29.4	12	70.6	17	11.2
Total Science	21	14	131	86	152	100

Table 4

Total CT Approaches Examples from CT Survey

CT Approaches	Not Accurate		Accurate		Total	
	Count	%	Count	%	Count	%
Collaboration	1	10	9	90	10	8.1
Creating	1	4.8	20	95.2	21	16.9
Debugging	3	17.6	14	82.4	17	13.7
Perseverance	2	33.3	4	66.7	6	4.8
Tinkering	1	1.4	69	98.6	70	56.5
Total Science	8	6.5	116	93.5	124	100

Value-mapping

The DBIR group took part in several activities that involved the diverse perspectives and experiences of the group through a process called value-mapping. The goal of value-mapping is to create a common language based on multiple points of view when integrating CT into elementary education. Ryoo and Shea (2015) believe that educators and researchers bring different values, experiences, and languages to the table. Through collaboration in value-mapping, a shared investment in research questions and strategies can be developed. Value-mapping was conducted, and several visual models were created with shared language and meaning on computational thinking. A Wordle was created toward the beginning, middle, and end of the research study based on the words that came to the minds of the DBIR team when thinking about CT in elementary school (Figures 5, 6, and 7). These Wordles depict how the research team's thinking changed over time.

The first word collection was not completed until a few months into the project when the DBIR team was deep into research and standards crosswalk for CT. Other than the research team who wrote the project proposal, only two other members of the DBIR team had any knowledge of CT at the start of the study. The next two-word collections occurred after much deliberation about what was happening in the classrooms. Over time, the number of words decreased in describing CT which indicates a clearer conceptual understanding of what CT entails. In addition, the word "problem" became smaller as time went on after the research team engaged in numerous conversations about the appropriateness of this terminology at the elementary level.

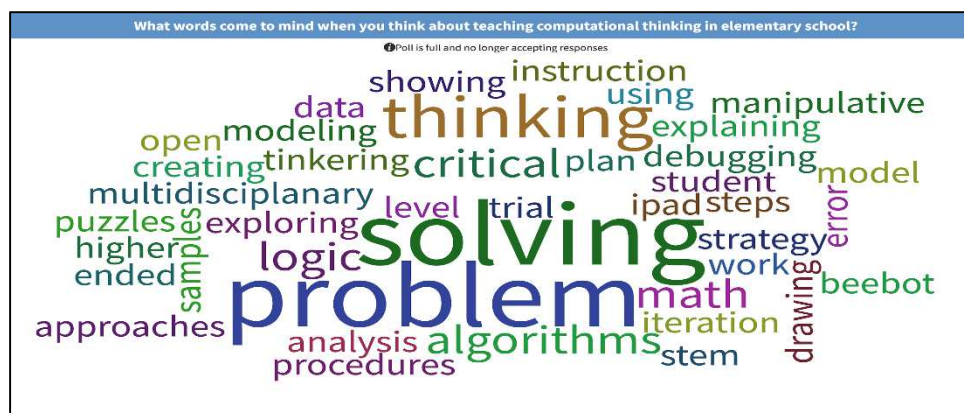


Figure 7

Value-Map of Wordle Created at Beginning of Research 2/7/19

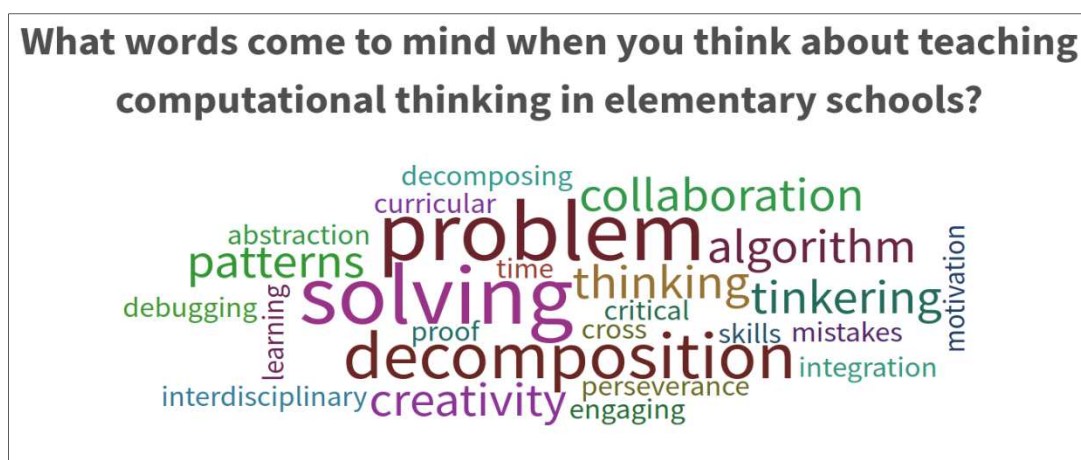


Figure 6

Value-Map of Wordle Created at Middle of Research 1/16/20

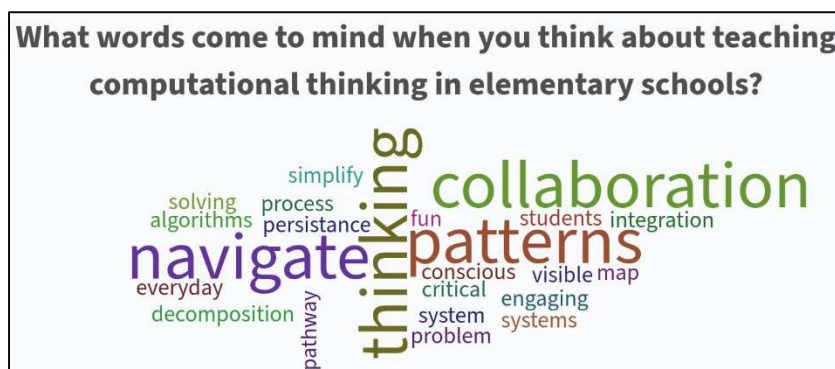


Figure 5

Value Map of Wordle Created at End of Research 4/13/21

Brainstorm for shared definition

In one of the last DBIR meetings, the research team revisited what CT meant by creating a Jamboard, which is a digital whiteboard created by Google that allows people to share ideas in real-time. Members used the sticky note feature to brainstorm words of what CT is and what it is not as shown in Appendix B. Once this was created, the group divided into three different breakout rooms to create a teacher-friendly definition that was shared on different slides in the Jamboard. From these three definitions a smaller group consisting of five members of the team synthesized these ideas to create the teacher-friendly definition that was presented to the larger group for member checking. The final definition for CT and the component parts are displayed in Table 5.

Table 5

Teacher-Friendly Definition of CT for Elementary Level

Computational Thinking - Logical thinking pathways that help to efficiently organize and identify relationships between concepts in order to make meaning	
CT tools to think about and develop meaning from the qualitative and quantitative data in our world	
algorithms	Organize and identify steps
decomposition	Identify and organize into parts
abstraction	Organize by removing non-important information
patterns	Identify and organize similarities

Discussion

After a critical review of standards and literature, value-mapping, designing, and analyzing survey data, challenging discussions, and three years of working together and participating in numerous activities around CT, the DBIR team came to understand CT as “logical thinking pathways that help to efficiently organize and identify relationships between concepts in order to make meaning.” The team looked at the concepts and approaches as tools to think about and develop meaning from the qualitative and quantitative data in our world. When the DBIR team first got together, only the educational researchers had a surface-level understanding of computational thinking, but by the end of the research study many members’ thinking and experiences with CT had changed (Pietros & Sweetman, 2020).

Throughout the three-year process, the thinking of the DBIR team evolved, which is evident in the value-mapping Wordles. The Wordles started out with numerous words used to describe computational thinking, but as time went on the number of words decreased and the ideas of CT became more refined. The ideas around solving problems also began to shift as indicated by the changing size of the words “solving” and “problems” in all three Wordles.

Analysis of the open responses collected on the survey data showed some accurate examples of the different CT concepts and approaches, which reinforces the idea that if teachers are given a teacher-friendly definition, they can accurately describe where that concept or approach is present in their instruction. For example, research from the survey indicated teachers can provide accurate examples of how CT integrates into science instruction. The concept *decomposition* was described as “the process of breaking

down a task into more manageable parts,” with examples given and a picture depicting the various parts that make up a flower. This led to 75 teacher examples with 97.3% accuracy deemed by the research team. However, it also brought attention to concepts teachers have a tough time understanding and finding examples of in their instruction, such as *abstraction*. This led the group to refine the definitions used in the survey.

The literature review helped create the teacher-friendly definition by using some of the language used in the variety of definitions while avoiding the terminology of problem-solving. Reviewing the literature also helped to narrow down the components of CT that should be used at the elementary level within science instruction. The concepts chosen as the main elements are the most frequent components used within the search conducted, which includes *abstraction*, *algorithms*, and *decomposition*. *Patterns* is the fourth concept used in the teacher-friendly definition of CT because it addresses the importance of using data while at the same time showing overlap with the cross-cutting concept of patterns used in the Next Generation Science Standards (NGSS, 2013).

Numerous conversations occurred throughout the three-year process around the words “problem-solving.” The DBIR team determined the phrase “problem-solving,” which appeared in most current definitions, could be problematic for elementary teachers. First, the term did not have the same meaning for all the members in the DBIR group. The term is used in elementary school for a variety of reasons, e.g., to address the social emotional challenges that occur on the playground. Second, the term “problem-solving” in the definitions implies that children only engage in CT practices when they have problems to be solved. Many of the teachers on the team viewed “problem” in a way similar to Merriam-Webster’s definition of “problem”: as “something that is difficult to

deal with: something that is a source of trouble, worry, etc.” (Merriam-Webster, n.d.) Most of the CT-integrated teaching activities that teachers reported were not problem-based, as evident in the Padlet shared by members of the team. For example, the Padlet had examples of algorithms showing the steps involved to plant seeds so they can learn how plants change and grow over time, or the steps needed in separating a mixture into its component parts. In addition, the team was concerned the term “problem-solving” could cause elementary educators to overwhelm young children with “adult” problems or to create superficial problems for activity’s sake rather than meaningful authentic learning experiences. With this thinking the team made a conscious effort to find different wording when creating a teacher-friendly definition.

Although the research team has not evaluated this new definition yet in the field, we believe these definitions are more teacher-friendly and useful for many reasons. One reason is that the definitions created are simple and were designed with input from K–5 teachers. When having discussions as a research team, the K–5 teachers unanimously affirmed that the simpler the definition the better, and the created definition is quite simple. The definitions were also created with leaders from a broad array of perspectives and backgrounds (researchers and higher-ed faculty, K–12 STEM+C leaders, school-based leaders, teachers, and the informal science community) and who have a strong interest in curriculum development and the implementation of CT. The different stakeholders were able to come to consensus on a definition which is not an easy task with such a diverse group. Another reason we believe the definitions for CT are more teacher-friendly for elementary science is that the problem-solving language was replaced with “efficiency” and “thinking pathways” to help describe what is happening in the

mind of elementary students before executing the thinking process, which is more appropriate for these younger learners. We do not want children in their earlier years to feel like the world is full of problems to be troubled and worried about, but rather positive activities and experiences that could potentially lead to a more diverse student population interested in pursuing careers in STEM+C fields.

The next steps for this research would include providing teachers with professional development to support identifying where CT is already happening or has the potential to happen in their instruction by creating professional learning communities, summer institutes, or mentor opportunities offered by teacher or industry leaders who have experience with CT. Using the new teacher-friendly created definitions will help bring awareness to teachers that integrating CT into their instruction is not adding one more task to their busy plates but creating an opportunity for their students to utilize thinking skills necessary for the 21st century. Teachers will be able to make their students' thought process more visible by pointing out and naming the diverse ways their students organize their thinking. This research will be helpful for curriculum designers, policy makers, and teachers by giving a common language for developing programs at the elementary level that will lay the foundation for an essential component needed in achieving computer science for all.

References

- Aho, A. V. (2012). Computation and Computational Thinking. *Computer Journal*, 55(7), 832–835. <https://doi.org/10.1093/comjnl/bxs074>
- 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. *Journal of Educational Technology & Society*, 19(3), 47-57
- Barefoot Computing. (n.d.). Retrieved from <https://www.barefootcomputing.org>
- 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/10.1145/2676723.2693616>
- Berland, M., & Wilensky, U. (2015). Comparing Virtual and Physical Robotics Environments for Supporting Complex Systems and Computational Thinking. *Journal of Science Education and Technology*, 24(5), 628–647. <https://doi.org/10.1007/s10956-015-9552-x>
- Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in the early childhood classroom*. Routledge.
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education-Implications for policy and practice* (No. JRC104188). Joint Research Centre (Seville site).
- Bocconi, S., Chiocciariello, A., & Earp, J. (2018). The Nordic approach to introducing Computational Thinking and programming in compulsory education. *Report prepared for the Nordic@ BETT2018 Steering Group*, 397-400.
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada* (Vol. 1, p. 25).
- Breslyn, W., & McGinnis, J. R. (2019). Investigating preservice elementary science teachers' understanding of climate change from a computational thinking systems perspective. *Eurasia Journal of Mathematics, Science and Technology Education*, 15(6) doi:10.29333/ejmste/103566
- Cateté, V., Lytle, N., Dong, Y., Boulden, D., Akram, B., Houchins, J., Barnes, T., Wiebe, E., Lester, J., Mott, B. & Boyer, K. (2018). Infusing computational thinking into middle grade science classrooms: lessons learned. In *Proceedings of the 13th Workshop in Primary and Secondary Computing Education* (pp. 1-6).
- Code.org, CSTA, & ECEP Alliance. (2020). 2020 State of Computer Science Education: Illuminating Disparities. Retrieved from <https://advocacy.code.org/stateofcs>

- Computer Science Teachers Association (2020). *Standards for Computer Science Teachers*. Retrieved from <https://csteachers.org/teacherstandards>
- Computing At School. (2013). Computing in the national curriculum: A guide for primary teachers. Belford, UK: Newnorth Print. Retrieved from <http://www.computingatschool.org.uk/datauploads/CASPrimaryComputing>
- Creswell, J. W. (2014). The selection of a research approach. *Research design: Qualitative, quantitative, and mixed methods approaches*, 3-24
- CS for All (2021). Computer Science for All. Retrieved October 26,2021, from <https://www.csforall.org>
- Cuny, J., Snyder, L., Wing, J.M. (2010). Demystifying computational thinking for non-computer scientists. Unpublished manuscript, referenced in <http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf>.
- Denning, P. J., & Tedre, M. (2019). *Computational Thinking*. MIT Press.Ehsan, H., Rehmat, A. P., & Cardella, M. E. (2021). Computational thinking embedded in engineering design: capturing computational thinking of children in an informal engineering design activity. *International Journal of Technology & Design Education*, 31(3), 441–464. <https://doi-org.uri.idm.oclc.org/10.1007/s10798-020-09562-5>
- Gane, B. D., Israel, M., Elagha, N., Yan, W., Luo, F., & Pellegrino, J. W. (2021). Design and validation of learning trajectory-based assessments for computational thinking in upper elementary grades. *Computer Science Education*, 31(2), 141–168. <https://doi-org.uri.idm.oclc.org/10.1080/08993408.2021.1874221>
- Google Inc. & Gallup Inc. (2017, December). Encouraging Students Toward Computer Science Learning. Results From the 2015-2016 Google-Gallup Study of Computer Science in the U.S. K–12 Schools (Issue Brief No. 5). Retrieved from <https://goo.gl/iM5g3A>.
- Grover, S. (2018). The 5th ‘C’ of 21st century skills. *Try computational thinking (not coding)*. (March 13). Retrieved from EdSurge News: <https://www.edsurge.com/news/2018-02-25-the-5th-c-of-21st-century-skills-try-computational-thinking-not-coding>.
- Grover, S., & Pea, R. (2018). Computational thinking: a competency whose time has come. In S. Sentance, E. Barendsen, & S. Carsten (Eds.), *Computer science education: perspectives on teaching and learning in school* (pp. 19–37). London: Bloomsbury Academic.
- Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, 42(1), 38-43.
- Guba, E. G. & Lincoln, Y.S. (1994). Competing paradigms in qualitative research. In N. Denzin & Lincoln, Y.S., *Handbook of qualitative research* (1st ed. 1994). (pp.105-117). Thousand Oaks, CA: Sage Publications, Inc.

- Henderson, P. B., Cortina, T. J., & Wing, J. M. (2007). Computational thinking. *ACM SIGCSE Bulletin*, 39(1), 195-196.
- ISTE, CSTA. (2011). *Computational thinking in K–12 education leadership toolkit*.
- ISTE (2018). ISTE Standards for Educators: Computational Thinking Competencies. (International Society for Technology in Education). <https://www.iste.org>.
- K–12 Computer Science Framework (2016). *K–12 computer science framework*. <https://www.k12cs.org/>
- Kang, E. J. S., Donovan, C., & McCarthy, M. J. (2018). Exploring Elementary Teachers' Pedagogical Content Knowledge and Confidence in Implementing the NGSS Science and Engineering Practices. *Journal of Science Teacher Education*, 29(1), 9–29. <https://doi-org.uri.idm.oclc.org/10.1080/1046560X.2017.1415616>
- Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 4(3), 583-596.
- Karpinski, Z., Biagi, F., & Di Pietro, G. (2021). Computational Thinking, Socioeconomic Gaps, and Policy Implications. IEA Compass: Briefs in Education. Number 12. *International Association for the Evaluation of Educational Achievement*.
- Kaya, E., Newley, A., Yesilyurt, E., & Deniz, H. (2020). Measuring Computational Thinking Teaching Efficacy Beliefs of Preservice Elementary Teachers. *Journal of College Science Teaching*, 49(6), 55–64.
- 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), 174–188. <https://doi.org/10.1007/s10956-019-09798-4>
- Koehler, M., & Mishra, P. (2009). What is technological pedagogical content knowledge (TPACK)? *Contemporary Issues in Technology and Teacher Education*, 9(1), 60-70.
- Maltese, A. V., & Tai, R. H. (2010). Eyeballs in the fridge: Sources of early interest in science. *International Journal of Science Education*, 32(5), 669-685. <https://doi-org.uri.idm.oclc.org/10.1080/09500690902792385>
- Martínez-García, E. (2021). Computational thinking: the road to success in education. *Academian Letters*, Article 3973. <https://doi.org/10.20935/AL3973>.
- Massachusetts Department of Elementary and Secondary Education. (2016). *Massachusetts Digital Literacy and Computer Science Curriculum Framework*. Retrieved October 31, 2021, from <http://www.doe.mass.edu/stem/dlcs/>
- Merriam-Webster. (n.d.) Merriam-Webster.com dictionary. Retrieved November 26, 2021, From <https://www.merriam-webster.com/>

- McGinnis, J. R., Ketelhut, D. J., Mills, K., Hestness, E., Jeong, H., & Cabrera, L. (2019). Preservice Science Teachers' Intentions and Avoidances to Integrate Computational Thinking into Their Science Lesson Plans for Young Learners. *Grantee Submission*.
- NGSS Lead States. (2013). *Next generation science standards: For states, by states*. Washington, D.C.: National Academies Press.
- National Research Council. (2012). *A framework for K–12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press.
- Ogegbo, A. A., & Ramnarain, U. (2021). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, 1–28. <https://doi-org.uri.idm.oclc.org/10.1080/03057267.2021.1963580>
- Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, 1(1), 95-123.
- Patton, M. Q. (2015). *Qualitative research & evaluation methods: integrating theory and practice*. Thousand Oaks, CA: SAGE Publications, Inc.
- Pietros, J. & Sweetman, S. (2020). The importance of research practice partnerships for professional development. *Journal of the National Association for Professional Development Schools*, Special Edition Partners: Bridging Research to Practice, 15(2) 23-25.
- Rhode Island K–12 Computer Science Education Standards. (2018). *Rhode Island K–12 Computer Science Education Standards*. Retrieved November 27, 2021, From https://www.ride.ri.gov/Portals/0/Uploads/Documents/Instruction-and-Assessment-World-Class-Standards/Other-Subjects/RI_CS_Ed_Standards_May2018.pdf
- Rich, K. M., Yadav, A., & Schwarz, C. V. (2019). Computational Thinking, Mathematics, and Science: Elementary Teachers' Perspectives on Integration. *Journal of Technology & Teacher Education*, 27(4), 165–205
- Rich, K. M., Yadav, A., & Larimore, R. A. (2020). Teacher Implementation Profiles for Integrating Computational Thinking into Elementary Mathematics and Science Instruction. *Education and Information Technologies*, 25(4), 3161–3188
- Ryoo, J & Shea, M. (2015) Value mapping: An activity for surfacing power dynamics and diverse perspectives in research-practice collaborations. *Research Practice Collaboratory*.
- 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>
- Sweetman, S. (Principal Investigator). (2018-2021). *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (Award number: 1813224) [Grant] National Science Foundation.

- Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 20(4), 715-728.
- Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1), 127-147. <https://doi.org/10.1007/s10956-015-9581-5>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
- Wing, J. (2011). Research notebook: Computational thinking—What and why. *The link magazine*, 6, 20-23.
- Yadav, A., Mayfield, C., Zhou, N., Hambruch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, 14(1), 1-16.

Appendix A

References Used in Literature Review

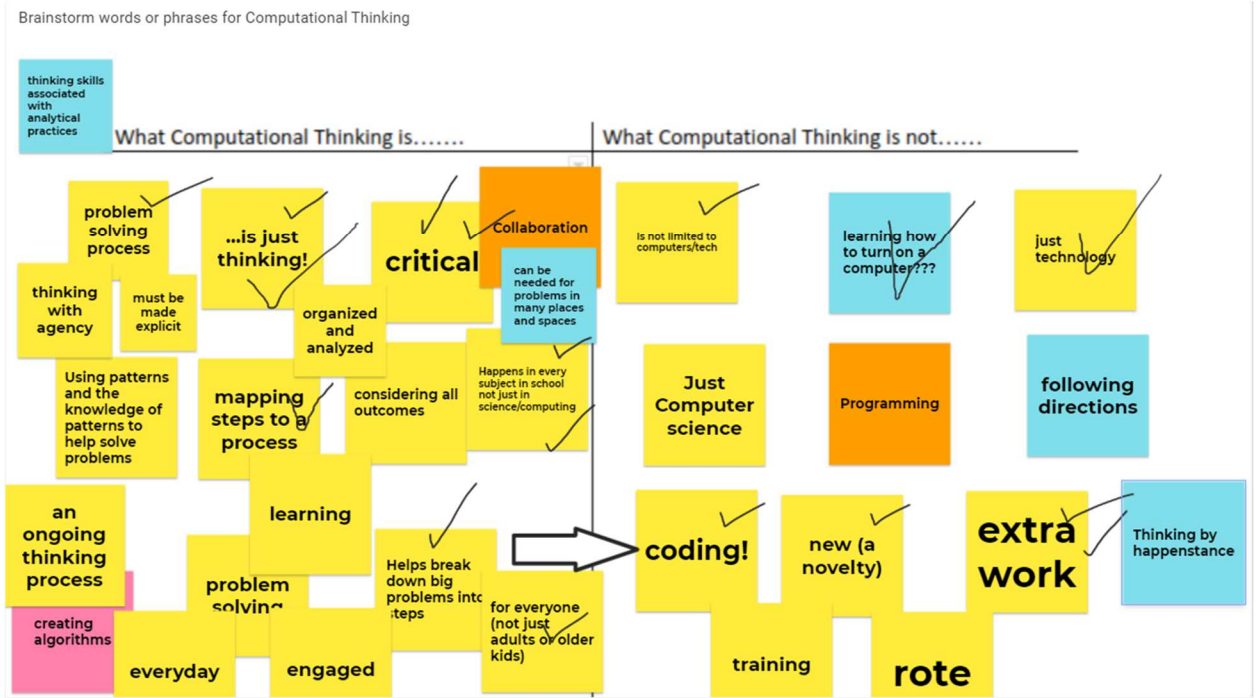
- Breslyn, W., & McGinnis, J. R. (2019). Investigating preservice elementary science teachers' understanding of climate change from a computational thinking systems perspective. *Eurasia Journal of Mathematics, Science and Technology Education*, 15(6) doi:10.29333/ejmste/103566
- Dickes, A. C., Farris, A. V., & Sengupta, P. (2020). Sociomathematical Norms for Integrating Coding and Modeling with Elementary Science: A Dialogical Approach. *Journal of Science Education & Technology*, 29(1), 35–52. <https://doi-org.uri.idm.oclc.org/10.1007/s10956-019-09795-7>
- Ehsan, H., Rehmat, A. P., & Cardella, M. E. (2021). Computational thinking embedded in engineering design: capturing computational thinking of children in an informal engineering design activity. *International Journal of Technology & Design Education*, 31(3), 441–464. <https://doi-org.uri.idm.oclc.org/10.1007/s10798-020-09562-5>
- Ehsan, H., Rehmat, A. P., & Cardella, M. E. (2019). Computer Science Unplugged. *Science and Children*, 57(3), 56–62. <https://doi-org.uri.idm.oclc.org/10.1080/08993408.2021.1874221>
- Hestness, E., Ketelhut, D. J., McGinnis, J. R., & Plane, J. (2018). Professional Knowledge Building within an Elementary Teacher Professional Development Experience on Computational Thinking in Science Education. *Journal of Technology & Teacher Education*, 26(3), 411–435.
- Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, 82, 263–279. <https://doi-org.uri.idm.oclc.org/10.1016/j.compedu.2014.11.022>
- Jaipal-Jamani, K., & Angeli, C. (2017). Effect of Robotics on Elementary Preservice Teachers' Self-Efficacy, Science Learning, and Computational Thinking. *Journal of Science Education and Technology*, 26(2), 175–192. <https://doi.org/10.1007/s10956-016-9663-z>
- Kang, E. J. S., Donovan, C., & McCarthy, M. J. (2018). Exploring Elementary Teachers' Pedagogical Content Knowledge and Confidence in Implementing the NGSS Science and Engineering Practices. *Journal of Science Teacher Education*, 29(1), 9–29. <https://doi-org.uri.idm.oclc.org/10.1080/1046560X.2017.1415616>
- Kaya, E., Yesilyurt, E., Newley, A., & Deniz, H. (2019). Examining the Impact of a Computational Thinking Intervention on Pre-service Elementary Science Teachers'

- Computational Thinking Teaching Efficacy Beliefs, Interest and Confidence. *Journal of Computers in Mathematics & Science Teaching*, 38(4), 385–392.
- Kaya, E., Newley, A., Yesilyurt, E., & Deniz, H. (2020). Measuring Computational Thinking Teaching Efficacy Beliefs of Preservice Elementary Teachers. *Journal of College Science Teaching*, 49(6), 55–64.
- 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), 174–188. <https://doi.org/10.1007/s10956-019-09798-4>
- Luo, F., Antonenko, P. D., & Davis, E. C. (2020). Exploring the evolution of two girls' conceptions and practices in computational thinking in science. *Computers and Education*, 146. doi:10.1016/j.compedu.2019.103759
- McGinnis, J. R., Hestness, E., Mills, K., Ketelhut, D. J., Cabrera, L., & Jeong, H. (2020). Preservice Science Teachers' Beliefs About Computational Thinking Following a Curricular Module Within an Elementary Science Methods Course. *Contemporary Issues in Technology & Teacher Education*, 20(1), 1.
- McGinnis, J. R., Ketelhut, D. J., Mills, K., Hestness, E., Jeong, H., & Cabrera, L. (2019). Preservice Science Teachers' Intentions and Avoidances to Integrate Computational Thinking into Their Science Lesson Plans for Young Learners. *Grantee Submission*.
- Moore, T. J., Brophy, S. P., Tank, K. M., Lopez, R. D., Johnston, A. C., Hynes, M. M., & Gajdzik, E. (2020). Multiple representations in computational thinking tasks: A clinical study of second-grade students. *Journal of Science Education and Technology*, 29(1), 19-34. doi:10.1007/s10956-020-09812-0
- Ogebo, A. A., & Ramnarain, U. (2021). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, 1–28. <https://doi-org.uri.idm.oclc.org/10.1080/03057267.2021.1963580>
- Rich, K. M., Yadav, A., & Larimore, R. A. (2020). Teacher Implementation Profiles for Integrating Computational Thinking into Elementary Mathematics and Science Instruction. *Education and Information Technologies*, 25(4), 3161–3188.
- Rich, K. M., Yadav, A., & Schwarz, C. V. (2019). Computational Thinking, Mathematics, and Science: Elementary Teachers' Perspectives on Integration. *Journal of Technology & Teacher Education*, 27(4), 165–205.
- 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(1), 53-64. <https://doi.org/10.1007/s10956-019-09801-y>

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.uri.idm.oclc.org/10.1080/08993408.2018.1560550>

Appendix B

Brainstorm of What CT Is and Is Not



CHAPTER 3

PREDICTING COMPUTATIONAL THINKING IN ELEMNTARY SCIENCE USING A MULTILEVEL MODEL APPROACH

Currently in preparation to *Journal of Science Education and Technology*

Department of Education, University of Rhode Island
Kingston, Rhode Island, 02881

Abstract

Computational Thinking (CT) is an essential 21st-century problem-solving skill that is receiving increased attention in the educational field. Researchers and policy leaders have identified the need to integrate CT at the elementary level into core subjects to provide equitable access for all students. Science is a meaningful subject with many real-world, relevant problems into which CT can be applied. By assessing the current landscape of where CT concepts and approaches integrate authentically into science lessons, policymakers and district leaders can be more intentional in supporting implementation efforts. This research used an exploratory survey design to examine the frequencies of CT concepts (decomposition, algorithms, abstraction, and pattern recognition) and approaches (tinkering, creating, debugging, perseverance, collaboration) that exist in science in K–5 schools in a northeast state as reported by teachers (n=259). Hierarchical Linear Modeling (HLM) was used to analyze the influence of teacher and district factors on CT frequency. Over 90% of the variance in CT concepts and approaches lay in teacher attitudes and backgrounds and less than 10% lay among districts. Teacher-level factors including experience, grade level, and confidence significantly influence CT in the elementary classrooms as well as the district-level factor of being part of a Research-Practice Partnership. This study contributes to a better understanding of variables affecting CT teaching frequency that can be leveraged to impact reform efforts supporting CT integration in science.

Keywords: computational thinking, elementary science, integration, research practice partnership, hierarchical linear modeling

Introduction and Statement of the Problem

Teaching and learning in a rapidly changing society where technology is ubiquitous pose many challenges in education. As computers become more pervasive in children's everyday life, a high-quality science, technology, engineering, and math (STEM) education, with a focus on computational thinking skills, is needed. Students should learn how to collect, organize, analyze, and apply data to work through complex problems such as climate change, poverty, diseases, shortage of resources, and biodiversity losses (NRC, 2012; Rischard, 2007). STEM-related employment in the United States, which also includes computer science, is growing at a fast pace and is predicted to grow 15 percent from 2019 to 2029 (Bureau of Labor Statistics, 2021; Noonan, 2017; White & Shakibnia, 2019). Therefore, there is a need to increase access to computer science subject matter for every child; not only does it address the needs of the workforce and skills needed in the digital age, but more importantly, it addresses foundational educational needs such as data analysis, modeling skills, and being able to think critically to solve problems (Khine, 2018; Papert, 1980).

One key element that is essential in computer science is computational thinking (CT). Jeannette Wing introduced this process in 2006 in her seminal paper in which she described CT as “a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytic ability” (Wing, 2006, p. 33). Although there are multiple definitions for this term and ideas on how CT needs to be learned, the one this paper will use is

a problem-solving process that includes a number of characteristics and dispositions. CT is essential to the development of computer applications, but it

can also be used to support problem-solving across all disciplines, including the humanities, math, and science. Students who learn CT across the curriculum can begin to see a relationship between academic subjects, as well as between life inside and outside of the classroom. (Computational Thinking for Educators, 2015, p. 1)

This problem-solving process can be categorized into elements commonly used by researchers and policymakers including concepts and approaches based on the classification of Barefoot Computing at School and International Society for Technology Education (ISTE). The concepts, which form the building blocks of computer science, consist of *creating algorithms*, *using abstraction*, *using decomposition*, and *recognizing patterns* (Berry 2013; Prottsman, 2019). The approaches to learning or activities involved in attaining the concepts consist of *tinkering*, *creating*, *debugging*, *persevering*, and *collaborating* (Berry, 2013).

Exposing children to CT at an early age has motivated them to develop computer science skills (Baek et al., 2019; Bers et al., 2014; Usengül & Bahçeci, 2020). However, implementation of CT at the elementary level must consider the needs of districts and teachers for engagement to be effective. School systems need resources to effectively integrate the concepts and approaches into their epistemology, content, pedagogy, and practice (Bocconi et al., 2016). Currently, there is no set curriculum implemented in most school systems nor best practices established to educate teachers on how to do this effectively (Butler & Leahy, 2021). To make this happen, systematic research is needed before implementation can take place and be sustainable.

The purpose of this exploratory study is two-fold: first to examine the frequency of CT concepts and approaches currently integrated into elementary science classrooms; second, to identify whether teacher factors (grade level, teaching experience, professional development, confidence, and level of concern) or district-level factors (classification, socioeconomic status, and science curriculum) affect CT frequency levels, in aims to help with future implementation efforts. This study builds on the foundation for research and implementation of CT integration into science by addressing the following research questions:

RQ1 - How often are Computational Thinking (CT) concepts and approaches currently taught in K–5 science classrooms?

RQ2- To what extent are teacher characteristics, such as grade level, teaching experience, professional development, confidence, and level of concern for implementing CT, related to the frequency of teaching computational thinking concepts and approaches in K–5 science classrooms?

RQ3- To what extent are district characteristics such as district classification, socioeconomic status, and participation in a Research Practice Partnership (RPP) using the Guiding Education in Math and Science Network (GEMS-Net) science curriculum related to the frequency of teaching computational thinking concepts and approaches in K–5 science classrooms?

Theoretical and Conceptual Frameworks

To help integrate CT into the core curriculum, this study adopts two frameworks: CT-TPACK (Computational Thinking Technological Pedagogical Content Knowledge)

(Angeli et al., 2016; Koehler & Mishra, 2009) and Ecological Systems Theory (Bronfenbrenner, 1979).

A contemporary framework that should be considered when integrating CT into the teaching of science involves understanding a body of knowledge called TPACK, which stands for technology, pedagogy, content knowledge, and the interaction between and among them (Koehler & Mishra, 2009). Koehler & Mishra (2009) contend

“integration efforts should be creatively designed or structured for the particular subject matter” (p. 62). When integrating with science, content knowledge (CK) consists of scientific facts, theories, scientific methods, and evidence-based reasoning. Pedagogical knowledge (PK) is the teachers’ knowledge of processes and practices for teaching and learning, including understanding students’ cognitive and social development.

Pedagogical content knowledge (PCK) is understanding the type of pedagogy that works best with specific content. For example, in science, inquiry and problem-based learning are often used. Technology knowledge (TK) is constantly changing but refers to

knowledge of information technology for information processing, communication, and problem-solving. Technological Content Knowledge (TCK) is understanding how

technology and content influence and constrain one another. When integrating with science, there is Technological Pedagogical Content Knowledge (TPACK) that looks at

knowledge for advancing student learning and understanding through using different technological tools and different practices and teaching methods. TPACK is important for

teaching science and educational programs because it provides teachers a framework to consider when delivering instruction to their students and helps with implementation and

reform efforts for the twenty-first century by understanding how to develop their instructional knowledge (Hsu, 2015).

Angeli et al. (2016) contend TPACK—or slightly altered TPCK_{CT}—is an important conceptual framework to use when teaching computational thinking skills, which also includes context (CX_{CT}) and learner knowledge (LK_{CT}). The context refers to the scope, the micro-, mezzo-, and macro-level, whereas learner knowledge includes learner's difficulties with developing abstraction, identifying patterns, decomposing problems, and thinking algorithmically. The content knowledge (CK_{CT}) includes understanding the different concepts (algorithms, decomposition, abstraction, and pattern recognition) and approaches (tinkering, debugging, perseverance, collaboration, creating) of CT. Pedagogy knowledge (PK_{CT}) is an understanding of how to best teach problem-solving and knowledge for technology (TK_{CT}), including knowing when to use the correct tools to carry out the problem-solving process, whether they are plugged or unplugged.

The TPACK or TPCK_{CT} acronym can be modified to create a model called CT-TPACK when CT is integrated into teaching science class rather than CT taught as stand-alone instruction (Figure 1). In addition to having knowledge about science content, pedagogy, and technology, the teacher needs to know about how CT will enhance their teaching and the understanding of the content by the students. Teachers need to understand different teaching practices that are conducive to the integration of the thinking processes for solving problems such as load-reducing, schema-activation, structured-based, generative, guided-discovery, modeling, and teaching thinking skills (Kale et al., 2018). Overall, the teachers need to understand the correct tools, practices,

and content when teaching students how to solve problems using computational thinking skills in the field of science.

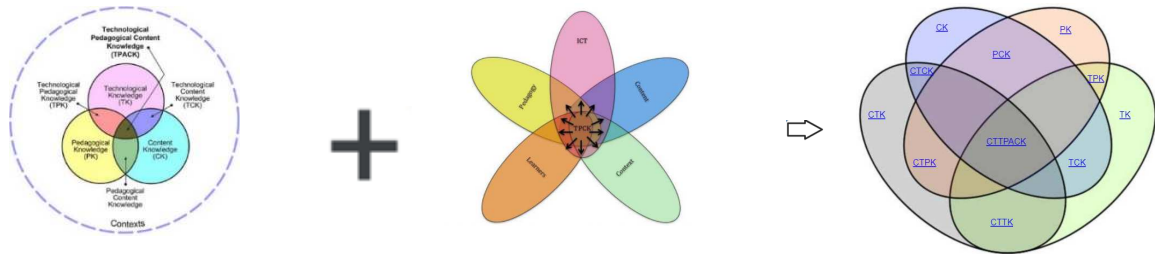


Figure 1

CT-TPACK Model. This model combines ideas from Koehler & Mishra (2009) and Angeli et al. (2016).

Another theory that guides this research is Urie Bronfenbrenner's Ecological Systems Theory. According to Bronfenbrenner (1979), different layers or systems in a child's environment influence their development and learning. These environments are nested within one another in different layers that interact directly and indirectly. The layers consist of the microsystem, the mesosystem, the exosystem, and the chronological layer. The first layer closest to the child is the microsystem, which is the immediate environment. The microsystem incorporates people who directly influence the child, such as teachers and parents. The mesosystem is the second layer and consists of the relationships between the different structures in the microsystem and how they affect the child. The next layer is the exosystem, the indirect environment that can influence the child's development, such as district-level personnel who make curricular decisions. The

outermost layer, the macrosystem, includes social and cultural values that influence the inner layer systems, such as socioeconomic status, ethnicity, and poverty. The chronological layer, the last layer, encompasses both external and internal changes in the child and includes changes such as major life and historical events.

The educational system and the ways the layers of the Ecological Systems Theory would need to interact to influence the integration of CT into science classrooms is complex. This study examines the microsystem and exosystem that exist in this multilayer context. The microsystem, or the qualities of the teacher that directly influence the students, will be examined to determine if variables such as experience, confidence, grade level, or level of concern affect the teaching of CT. The exosystem, or the indirect environment, will be examined to determine if variables such as location (rural, suburban, and urban), socioeconomic status, and district-level curricula decisions impact the use of CT in the science classroom.

A few recent studies have looked at these variables independently. In a small-scale study, Kale et al. (2018) found teachers in rural settings had significantly lower CT skills than teachers in urban environments. Karpinski et al. (2021) found that students from less advantaged backgrounds have fewer CT skills than students from more advantaged backgrounds. There are multiple studies on how curriculum impacts the implementation of CT that are described in the following literature review.

Bronfenbrenner's Ecological Systems Theory and TPACK (CT-TPACK) converge to frame this study. Bronfenbrenner's theory describes the different system levels that affect learning and development and the CT-TPACK conceptual framework

stresses the importance of understanding a body of knowledge teachers must possess for successful integration and implementation.

Literature Review

What is Computational Thinking?

The International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) have listed the following characteristics of computational thinking as:

Formulating problems in a way that enables us to use a computer and other tools to help solve them, logically organizing and analyzing data, representing data through abstractions, such as models and simulations, automating solutions through algorithmic thinking, identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources, and generalizing and transferring this problem-solving process to a wide variety of problems. (p.13)

The goal of CT is not for students to just think like a computer scientist but to be able to solve problems across all disciplines (Barr & Stephenson, 2011). CT can be broken down into concepts and approaches that are commonly used for teaching and learning. The concepts students need to develop to increase their CT skills include knowing “a set of rules to get something done” (algorithm), “breaking down a task into smaller, more manageable parts” (decomposition), “spotting similarities and common differences” (patterns) and “identifying what's important without worrying too much

about detail” (abstraction) (Barefoot Computing, n.d.). To develop the four concepts of CT, Barefoot Computing recommends students engage in the following five approaches: explore and experiment to try things out (tinkering), plan and make things (creating), find and fix errors through predicting what should happen, find out exactly what did happen, work out where something went wrong (debugging), be determined, resilient, and tenacious (persevering) and work with others to ensure the best results (collaborating) (Barefoot Computing, n.d.).

CT in elementary science

Computational thinking is best learned when embedded in class subjects and taught in context using an interdisciplinary approach (Grover, 2018). Past initiatives in computer science have paid more attention to addressing the college and secondary levels rather than the elementary level (Bart, 2015; Google Inc. & Gallup Inc., 2016; Kafura et al., 2018; Kalelioglu et al., 2016; Weinberg, 2013). However, efforts and attention are now being focused on the teaching and learning of CT in elementary classrooms by integrating it into core subjects such as science (Ketelhut et al., 2019; Lee et al., 2019, Yadav et al., 2017).

The rationale for integrating CT at the elementary level is to introduce key concepts and experiences to children at an early age to develop positive attitudes towards disciplines; it has influence in career choice and allows more access to foundational skills and higher-paying jobs (Karpinski et al., 2021; Maltese & Tai, 2010). Since these skills take time to develop, starting young is never too early; CT skills have been successfully taught to children as young as four (Bers, 2018; Bers et al., 2014). As found in reading

and math, educational interventions that begin early have stronger positive outcomes than interventions that start later (Mattera & Morris, 2017; McIntosh et al., 2006).

Children exposed to STEM curriculum and programming at an early age have fewer gender stereotypes when considering future careers (Metz, 2007). The Code.org Advocacy Coalition (2018) posits that females and underrepresented groups exposed to computer science early in their schooling are less likely to be influenced by negative cultural stereotypes. By leveling the playing field and allowing more children access to computer science at an early age, future innovations and problem-solving can be made by a more diverse population representative of all backgrounds.

The rationale for integrating CT into science classrooms is supported by the idea that computing concepts and practices have become a fundamental part of the work professional scientists do (Denning & Tedre, 2019). Integrating CT into science gives learners a more realistic view of how scientists conduct their research in the field. Also, science is a meaningful context with many real-world, relevant problems in which CT can be applied to help solve them. Weintrop et al. (2016) contend that “science and mathematics are meaningful contexts in which we can successfully situate the concepts and practices of computational thinking’ (p. 128) and address the issue of reaching all students by occurring in a core discipline. Additionally, Ketelhut et al. (2019) argue that integrating CT into the science curriculum results in a deeper understanding of science through strong inquiry and student-centered lessons. Barr and Stephenson (2011) believe the effective implementation of CT can enhance students’ data analysis and modeling skills. CT will also better prepare students for the future economy and citizenship (Yadav

et al., 2017). In addition, the Next Generation Science Standards includes CT as one of the eight engineering practices (NGSS Lead States, 2013).

Implementation challenges

Many school systems are facing challenges as they strive to implement CT in elementary science instruction. Research has highlighted a list of challenges, including the lack of teachers available with computational skills, the lack of money to hire teachers with the skills, the priority for classroom instructional time to be spent on subjects that are tested, the limited time for planning CT-infused lessons, the lack of funds to purchase materials for activities, and the overwhelming nature of CT integration especially when teachers are not confident, knowledgeable, or experienced with science or CT (Google and Gallup, 2016; Ketelhut et al., 2019). However, teaching science can sometimes be a challenge for elementary teachers because they lack the coursework and experience necessary to feel confident when teaching children scientific concepts and practices. They often are responsible for other subjects such as social studies, math, and ELA. Also, educators need to teach different disciplines that include life, earth and space, and physical science using reform-based scientific practices (Davis & Smithey, 2009). For implementation to be successful, support is needed by administrators and district leaders to help teachers. Only 18% of administrators reported that computer science was a top priority in their school in Grades 1–6 (Google, 2015).

Other challenges with implementation at the teacher level include the needs to increase awareness of CT, have support from administrators, have a shared definition, and develop professional development on how to integrate CT. Sands et al. (2018)

revealed that teachers have some conceptual ideas about what computational thinking is. Still, they also have some misconceptions, such as thinking using digital tools is considered CT. Awareness needs to happen before CT can even be implemented. Israel et al. (2015) believed it was important for administrators to not only be supportive, but also to understand how computing impacts pedagogy, classroom management, and evaluation of state assessments. Also, not everyone shares the same definition for CT, and there is no clear-cut way for CT to be positioned in the curriculum. It takes time to gather information and data on what the best method is for implementing this change. To form a shared understanding, the information must be collected on what researchers, practitioners, and policymakers think CT is. There is a need to figure out how to place CT in the curriculum, consider what is developmentally appropriate for different grade levels and how this applies in various contexts. There needs to be a vision with clear goals where all stakeholders are on board. A systematic roll-out needs to be considered for teacher professional development and suitable assessments. Lastly, this policy needs continued support to be sustainable and effective communication between all the people involved (Bocconi et al., 2016). Many factors need to be considered when integrating CT into elementary science classrooms. Two areas showing promising results are curriculum development and training in-service and pre-service teachers.

CT curriculum

There are many programs, both computer-based (building-block programming, tangible programming, digital game creation, and robotics) and unplugged, being designed to teach CT at the elementary level (Brackmann et al., 2017). Unplugged

activities do not use a computer, instead, teachers use curricula such as Thinkersmith, Code.org, and CS Unplugged (Angeli & Jaipal-Jamani, 2018). Many of these programs have been taught as stand-alone instruction accessed by some students; however, more emphasis is now being placed on programs and curricula that integrate into core subjects that can be easily accessed by all students.

Examples of CT integrated into the science curriculum are becoming more prevalent. Bers (2018) created models where CT is integrated into core subjects with children as young as four using a tangible program called KIBO, utilizing open-ended activities and an approach she refers to as a “playground.” Sengupta et al. (2013) used CT in Simulation and Modeling (CTSiM) to teach science lessons and found significant learning gains in pre- and post-tests. Iyer (2019) created a curriculum called Computer Masti that has been used to teach over 1.5 million students in India from K–12. Barefoot Computing at Schools from the United Kingdom has an integrated curriculum for science available online (Barefoot Computing, n.d.). The National Integrated Cyber Education Research Center has created an integrated science curriculum for third to fifth grade (Computational Thinking Curriculum, 2019). Peel et al. (2019) designed an unplugged CT unit to successfully teach an integrated unit on natural selection. Waterman et al. (2020) provided a framework for taking existing science lessons and enhancing and extending them using CT to support science learning. They shared an integrated lesson concerning populations and ecosystems where abstraction was used to create a model of the relationships between deer, wolves, and resources. This lesson was enhanced by using spreadsheets to graph the relationship and extended by creating a simulation that speeds up what the population looks like over a long period of time.

Teacher development

Designing professional development for pre-service and in-service teachers is essential for CT implementation and needs to consider many factors (Lee et al., 2019; Yadav et al., 2016; Yadav et al., 2017). Factors include the concern or buy-in of the teachers regarding implementation, the location of the school and grade level being taught, student demographics, and the confidence or self-efficacy of the teacher with computer science innovations. Hall et al. 2013 believe when starting the implementation process, it is best to determine the teachers' level of concern and address their concerns through various methods such as coaching, mentoring, or explaining in more detail the importance of the change. Addressing concerns increases teachers' buy-in to the change.

District location (such as urban or rural) and grade level also play a role in teachers' motivation, students' physical skills, and users' access to computational thinking (Kale et al., 2018). In addition, teachers educating students in less affluent districts might need additional support in their training. Karpinski et al. (2021) reported data from the 2018 International Computer and Information Literacy Study indicating students from less advantaged backgrounds have lower levels of CT skills than those from more advantaged backgrounds. Therefore, designing effective CT professional development to address this inequity is necessary. Research has also indicated professional development in CT helps build teachers' self-efficacy when integrating it into their teaching (Jaipal-Jamani & Angeli, 2017; Rich et al., 2017). The thoughtful design of professional development using modeling experiences helps increase teachers' confidence or self-efficacy by allowing teachers to believe in their ability to complete a task which gives them the confidence and skills to do it (Bandura, 1986). Factors that

contribute to being a confident science teacher include having a strong science background, having a desire to implement reform-based practices, and elementary science teaching experience (Maeng et al., 2020).

Recommendations on best practices for tools, concepts, and practices are becoming available from various research studies. Sands et al. (2018) examined K–12 in-service teachers' perceptions of CT in elementary classrooms and revealed that the teachers need first to understand what CT is before it becomes integrated into K–12 practices. They believe if problem-solving is used as the focus, more teachers will be motivated to embed CT into their disciplines. Ruberg and Owens (2017) also found using strategies that included focusing on the problem-solving process, incorporating programming tools, and a career vision led to positive results with increased test scores on the math state assessment for the whole district. Still, it took six years to get these results.

Success was attributed to embedded coaching and having supportive administrators in Israel et al.'s (2015) exploratory cross-case analysis on seven elementary teachers and two administrators. The teachers in this research study had limited computer science experience and used CT-integrated curriculum with diverse learners, including students with low socioeconomic status or disabilities. A coached lesson cycle and ongoing mentoring and support were effective strategies for Ouyang et al. (2018) in training 21 elementary teachers in California when they integrated CT into their science lessons in Grades 4–6 after two years of training. Adler & Kim (2018) found modeling and simulations were effective ways for pre-service teachers to learn science content and to be encouraged to use computational thinking in their future classrooms.

Adler & Kim used Scratch to program a model of the solar system and a web-based simulation to help teach Newton's Second Law of Motion. Hestness et al. (2018) performed a qualitative research study on thirteen mentor teachers, finding teachers were able to identify how elements of CT were already present in their lessons and that CT was good for helping with persistence and giving students more ideas for careers. Yadav et al.'s (2014) implementation of CT modules with pre-service teachers found they were more likely to integrate CT in their future classrooms than pre-service teachers who did not use the modules. Ketelhut et al. (2019) conducted a case study investigating how to best support mentor teachers' attempts at integrating CT into their elementary science lessons. Results indicated teachers were able to deepen their science learning when given the time, space, and support to discuss the lesson and their attempts at integrating CT into science activities.

Belonging to a research-practice partnership has also supported teachers in the classroom (Coburn et al., 2021). Research-Practice Partnerships (RPPs) are characterized as long-term collaborations between researchers and educators focused on continuous improvement and paying close attention to solving problems of practice (Coburn & Penuel, 2016). RPPs are a viable strategy for integrating CT in the science classroom where both researchers and practitioners benefit from the partnership. Researchers can test and develop instructional activities in a real-world environment. At the same time, practitioners can be provided opportunities for resources to investigate problems of interest to them and feel supported while taking risks in the classroom. RPPs provide opportunities for new voices, questions, and observations when conducting research (Mulvey et al., 2020; Penuel et al., 2013).

Boulden et al. (2019) experienced some success integrating CT into a middle school science classroom using an RPP. They found that best practice consisted of finding teachers willing to engage in the process and take risks. At the same time, researchers needed to be flexible with school schedules, meeting teachers at their comfort level, and making frequent visits to the classroom for support. They also suggested that both researchers and practitioners need to engage in timely and thoughtful communication to successfully integrate CT. Research has clearly shown progress is being made in integrating CT in science. These studies provide a foundation for further research to respond to the growing need for students to gain CT skills and concepts.

Research Design

In this research, teacher-reported survey data is used to better understand where CT concepts and approaches are already integrated into the science classrooms. The results can guide in-service science teacher professional development and curriculum development for integrating CT into what is already done in the classroom. This quantitative study is unique because most of the current research on the integration of CT in science is exploratory and mostly qualitative. Qualitative data can help provide rich insight into human behaviors and uncover emic views (Guba & Lincoln, 1994). However, the context is often unique, bounded by time, and the samples are too small for generalizations. This study is a quantitative study using multilevel design Hierarchical Linear Modeling (HLM), which allows for generalizing and replicating findings (Creswell, 2014). HLM also is more efficient at accounting for variance among variables at different levels than other existing analyses and thus allows us to consider impacts in the complex systems of education (Woltman et al., 2012).

Specifically, this study examines elementary teachers' frequency of integrating CT concepts and approaches during science instruction and its relationship to various ecological factors to get a good idea of the current landscape. The teacher variables of interest include the years of experience teaching, grade level taught, the amount of time spent engaging in computer science professional development, levels of confidence teaching CT, and teachers' levels of concern for implementing CT in the classroom. The district variables of interest include the district classification of the urban, urban ring, rural, or suburban; socioeconomic status based on the percentage of families below the poverty level; and whether the district uses the GEMS-Net curriculum, which utilizes FOSS kit-based materials and has ongoing progressive professional development while participating in the Research-Practice Partnership.

Methods

The research performed in this study involved using a portion of survey data that is part of a larger research STEM+C study entitled *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (NSF Award number: 1813224). The larger study investigates how teachers perceived CT within their current lessons across all core domains, math, science, social studies, and science. This study is an analysis of CT in science and the systematic influences that impact the outcomes.

Participants

Elementary school teachers in Grades K–5 in a northeast state were invited to participate in the Computational Thinking Survey as part of the larger STEM+C research

study. The teachers were invited through contact with their superintendents and principals via emails and individually through emails obtained from school websites based on a Department of Education database. Every school in the database was researched to find publicly listed emails, which resulted in a list of over three thousand teacher, principal, and superintendent emails representing all the schools. Every teacher on this list was sent an email to reduce coverage and sampling error. The total number of elementary teachers who took the survey was 560, with a response rate of 12% of the state's teacher population. Of the 560 teachers who responded, 259 teachers from 30 different districts taught science. The other 301 responses were not used because they were from teachers who do not teach elementary science but teach other disciplines such as ELA, math, social studies, specials, and special education. The demographics of the science teachers were 94.2% females, 94.6% white/Caucasian, and 77.7% having over ten years of teaching experience. These findings are like the demographics of the national elementary teacher population (Banilower et al., 2018).

The Social Exchange Theory was applied to increase responses and reduce non-responses to questions in surveys. This theory explains that “people are more likely to comply with a request from someone else if they believe and trust that the rewards for complying with that request will eventually exceed the costs of not complying” (Dillman et al., 2014, p. 24). Due to the educational nature of the survey, teachers who chose to take the survey were able to learn about computational thinking and receive the benefit of earning one Professional Learning Unit (PLU). A PLU is documentation of professional development where teachers improve, enhance, or increase their knowledge, skills, or effectiveness needed as a condition of their certification renewal (Teachers &

Administrators, n.d). There are 185 elementary schools in this northeast state with 62,499 students with approximately 4,500 teachers. The overall demographics of students being taught by these teachers are 8% African American, 3% Asian, 25% Hispanic, 4% Multiracial, 1% Native American, and 59% White. Of this population, 47.5% of the students are eligible for subsidized lunch, and 15% of the population receive special education services (Count, 2018). Only teachers who took the survey and taught science participated for this research.

Survey

The Computational Thinking Survey was created by university researchers using the Tailored Design Method (Dillman et al., 2014), and delivered via SurveyMonkey software. The CT survey was examined for face, construct, and content validity by a Design-Based Implementation Research Group composed of 25 professionals: elementary science and math coaches, elementary teachers, district curriculum coordinators, and University educators with STEM+C backgrounds. A test-retest procedure was implemented to determine reliability using Spearman's rank-order correlation. There was a statistically significant, strong positive correlation between the test and retest with a Spearman rho of 0.840, $p < .01$ (Laerd Statistics, 2018). The survey instrument was piloted with 125 elementary teachers from March to June 2019, and minor changes were made to improve the questions, format, and frequency scales. The survey was administered late fall of 2019 and winter of 2020.

The cross-sectional web-based survey required teachers to respond to fifty-five questions that asked them to report how often and what type of lessons they currently

teach that include CT practices. Teachers spent approximately 30 minutes completing both open- and closed-ended questions. Teacher-level and district-level demographics were collected. The survey was educative in describing the concepts (decomposition, abstraction, pattern recognition, algorithm) and approaches (perseverance, creativity, tinkering, debugging, collaboration) involved in CT using definitions, pictures, and examples of the different elements as described by Barefoot Computing at School curriculum. An example of a description of the CT concept of decomposition is shown in Figure 2.

Concept: Decomposition

"Decomposition is the process of breaking down a task into smaller, more-manageable parts. It has many advantages. It helps us manage large projects and makes the process of solving a complex problem less daunting and much easier to take on" (Barefoot Computing, n.d.). Whenever students are labelling, adding detail to concept maps, or creating instructions, life cycles, and timelines, they are practicing their decomposition skills. Solving a math problem, getting dressed for PE, planning a research project, or organizing a school event are other examples. (www.barefootcas.org.uk)

A flower is broken down into smaller parts.

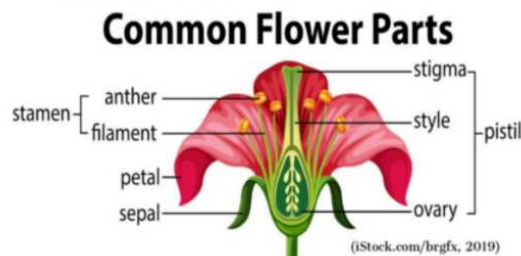


Figure 2

Example of CT Concept from Survey

All the different CT concepts and approaches were described in the same type of structure with a definition, examples, and a picture in the survey to establish a mutual understanding of these concepts and approaches across teachers. Survey questions

included the frequency levels of use for each concept and approach integrated into the science lessons.

One question, which appeared at the beginning and end of the survey, asked teachers their level of concern about implementing CT into their lessons based on the Stages of Concerns Based Adoption Model (CBAM) (Hord et al., 2013). The distinct levels of concern consist of seven stages:

- 1- "I've heard something about it, but other responsibilities take priority."
- 2- "This seems interesting, and I would like to know more about it."
- 3- "I am concerned about the changes I will need to make in my routines."
- 4- "I'm concerned about how much time it takes to get ready to teach with this new approach."
- 5- "How will this new approach impact my students?"
- 6- "I'm looking forward to sharing some ideas about it with other teachers."
- 7- "I incorporate computational thinking into my lessons now and have ideas about how to do it better."

Teachers chose which stage best aligned with their thinking. Confidence level in teaching CT was also asked at the beginning and end of the survey using a five-level Likert scale ranging from "not at all confident" to "extremely confident." In addition, questions were asked about hours of computer science professional development, location of the school, frequency of technology usage, teaching experience, and the type of curriculum used. The top five curricula reported included Stemscores, Gizmos, Kit-based, Make My Own, and GEMS-Net.

Previous data analysis using blockwise regression showed GEMS-Net as a predictor of CT use. Therefore, to further analyze this curriculum, a dichotomous variable was created for GEMS-Net. The GEMS-Net Research Practice Partnership uses FOSS kit-based materials and has ongoing progressive and mandatory professional development on research-based pedagogies. FOSS is an acronym for Full Option Science System which is

a research-based science curriculum for grades K–8 developed at the University of California. Belonging to an RPP is a critical component that differentiates the GEMS-Net curriculum from any other kit-based curriculum. Belonging to an RPP or not belonging to an RPP dichotomy was used because the emerging literature demonstrates promising insights for using a research-practice partnership framework when integrating CT (Boulden et al., 2018). A separate database was used to determine the percentage of families below the poverty level to serve as a proxy for socioeconomic status for the districts (National Center for Education Statistics, n.d.).

Variables of interest

The dependent variables of interest in this survey, which are the frequency or average minutes of CT concepts and approaches determined based on the total minutes of science taught each week multiplied by the percentage of lessons that integrated the different concepts and approaches. Independent variables were various teacher-level and district-level factors (Table 1).

Table 1

Teacher and District Level Variables for CT Frequencies

Level	Variables	Level	Variables
Teacher	Grade level (K–5) Teaching experience (years) Level of concern for CT Confidence level Professional development hours	District	Classification (rural, suburban, urban ring, urban core) Socioeconomic status (Percentage of families below the poverty level) Participation in a RPP using GEMS-Net curriculum

Results

Utilizing SPSS, the frequency of CT concepts and approaches were analyzed related to teacher-level and district-level factors. Teachers with more than four missing responses on the dependent variables of interest were excluded from the analysis.

Assumptions for linearity, normality, homoscedasticity, and independence of observations were analyzed. Since the data were nested and therefore the assumption of independence was violated, Hierarchical Linear Modeling (HLM) was used to analyze teachers (level-1) nested within districts (level-2) to allow for fewer assumptions from between and within-group differences (Raudenbush & Bryk, 2002). HLM is more efficient at accounting for variance among variables at different levels than other existing analyses (Woltman et al., 2012).

RQ1 - How often are Computational Thinking (CT) concepts and approaches currently taught in K–5 science classrooms?

Computational thinking concepts and approaches are currently being taught at different frequencies. Figure 3 shows the average minutes per week teachers spent teaching the different computational thinking concepts and approaches. The amount of time was calculated using the total minutes of science taught per week multiplied by the percentage of time teachers indicated teaching the different computational thinking concepts and approaches. Based on this data, the concept with the greatest average amount of minutes spent was decomposition with a mean of 89.75 minutes per week, followed by teaching patterns (78.32 min), algorithms (75.17 min), and abstraction (73.18 min). The approach with the most time spent was collaboration with a mean of 108

minutes per week. This was followed by perseverance (95.20 min), tinkering (89.58 min), creating (73.56 min), and lastly debugging (69.79). Time spent on collaboration was significantly more than debugging and creating.

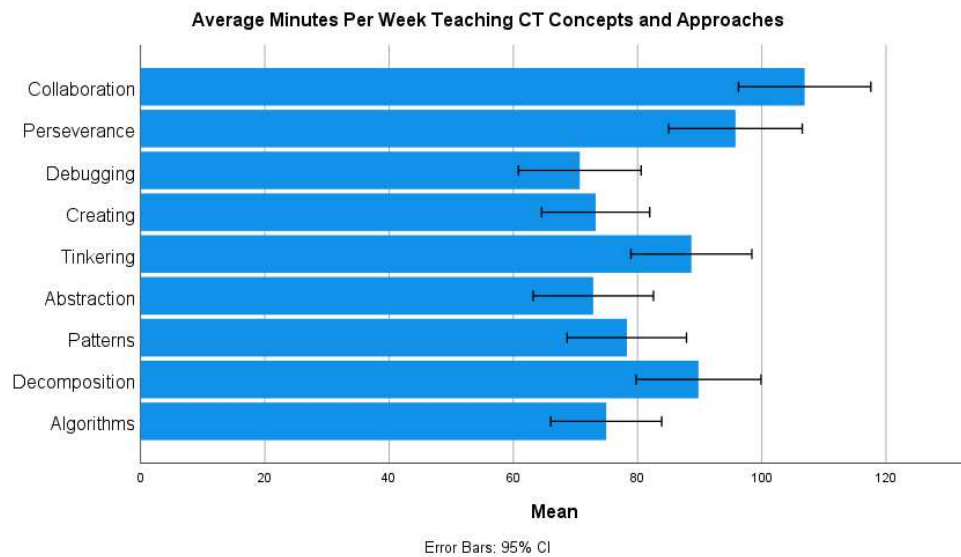


Figure 3

Average Minutes of Instruction Per Week Teaching CT Concepts and Approaches

Further analysis revealed 20% of teachers were teaching the different concepts and approaches for less than 30 minutes per week in comparison to the average teaching time of science of 151 minutes per week, with a range of 10 minutes to 450 minutes per week. This average teaching time is higher than the national average reported by the National Survey of Science and Math Education (NSSME) which is 20 minutes per day (Plumley, 2019). More than half of the teachers spent less than an hour per week teaching algorithms, patterns, abstraction, debugging, and creating. More than 25% of the teachers spent over two hours teaching students how to decompose problems. More than 25% of

teachers also taught lessons that included tinkering, persevering, and collaborating more than two hours a week. When analyzing the times of the different CT concepts and approaches it is important to know that they are not taught independently of one another but can be taught intertwined within lessons. This means creating and debugging can be happening at the same time as collaboration is happening in instruction.

RQ2- To what extent are teacher characteristics, such as grade level, teaching experience, confidence, professional development, and levels of concern for implementing CT, related to the frequency of teaching computational thinking concepts and approaches in K–5 science classrooms?

RQ3- To what extent are district characteristics such as classification, socioeconomic status, and science curriculum related to the frequency of teaching computational thinking concepts and approaches in K–5 science classrooms?

HLM was used to answer these two questions because it allows for partitioning the variation into within- and among-teachers from different districts (Raudenbush & Bryk, 2002). Of specific interest was the relationship between the computational thinking concepts and approaches (level-1 outcome variables) and both the teacher-level background and attitudes such as grade level, years of experience, stages of concern, confidence levels, and professional development frequencies (level-1 predictor variables) and their district classification, participation in a research-practice partnership, and district socioeconomic status (level-2 predictor variables).

When deciding on the most appropriate models, a series of steps and decisions were made. First, unconditional models were analyzed using an original data set

consisting of science teacher responses from thirty different districts collected for each outcome variable, consisting of total minutes per week of algorithms, decomposition, patterns, abstraction, tinkering, creating, debugging, perseverance, and collaboration. An unconditional model denotes the model with no predictors at either level. This model provides valuable information on the reliability of the slope estimates and the proportion of variance within and between districts. Out of 30 districts in the original data, only 15 districts had 5 or more participants. When the number of participants decreases, the reliability of the estimates deteriorates, resulting in biased estimates. It was decided to run the subsequent analyses using the 15 districts with five or more teachers per district (N=222).

Once the baseline unconditional models (Model 1 in Tables 2–10) were decided, the full saturated models were examined, allowing the slopes of all level-1 predictors (years of teaching experience, levels of concern, confidence, grade level, and professional development frequency) to be random. After examining the random slope models, it was decided to fix all the slope parameters as many slopes did not vary across districts (no significant random variations) and when there was significant random variation, allowing the slopes to be random makes the intercept parameters (mean differences between districts, which are of primary interests for this study) less reliable. The unconditional model and full model with fixed effects were run for the fifteen districts.

Among the five level-1 predictors, only three were statistically significant predictors of the outcomes: teacher experience, grade level, and teacher confidence levels. Levels of concern and professional development frequency were not significant factors for all nine dependent variables. To build the most parsimonious level-1 model, it

was decided to include only the three predictors. Once the most appropriate level-1 models (Model 2 in Tables 2-10) were decided, model building for level-2 predictors was considered. First, full level-2 models included district classification (2 dummy-coded variables for urban and urban rings as suburban districts were used as a reference group), socioeconomic status, and RPP curriculum. District classification variables turned out to not be significant predictors across all outcomes. The percentage of families below the poverty level was used as a proxy variable for socioeconomic status. RPP curriculum was included as a variable of interest because it is a district policy-related policy variable, whereas other predictors were not (context variables). In searching for the most appropriate and parsimonious level-2 models, a decision was made to exclude district classification variables, include the RPP curriculum, and present the results with and without percentages of families below the poverty level (Model 3 and Model 4 in Tables 2–10).

Model 1: $(CTMIN_{ij} = \gamma_{00} + u_{0j} + r_{ij})$

Tables 2-10 (unconditional models) shows that over 90% of the variance in computational thinking concepts and approaches lay within districts and less than 10% lay among districts, indicating most of the variances lie at the teacher level, ranging from 95% (debugging) to 90% (decomposition). Model 1 also serves as a baseline model against which the following models are compared in terms of variance explained in the model.

Model 2: $(CTMIN_{ij} = \beta_{0j} + \beta_{1j}*(EDUTIME_{ij}) + \beta_{2j}*(GRADELEV_{ij}) + \beta_{3j}*(CONFID2_{ij}) + r_{ij})$

Tables 2–10 includes the three teacher-level predictors: teaching experience, grade level, and confidence. The teaching experience was a significant predictor for the concepts of abstraction and recognizing patterns but not for algorithms or decomposition. The teaching experience was significant for the computational thinking approaches of debugging and collaboration but not for tinkering, creating, or persevering. Both grade level and confidence levels were significant for all the concepts and approaches. As grade levels increased from kindergarten to fifth grade, the time spent teaching the computational concepts and approaches increased, which aligns with children being able to participate in more problem-solving skills as they increase in age (Rijke et al., 2018). As confidence levels increased, so did time teaching concepts and approaches. Teachers who believe they can do something well are more likely to spend more time doing it than someone uncertain of their ability.

The explained variance in Model 2 was different for each of the CT concepts and approaches. Abstraction had the highest variance for the CT concepts explained by the teacher level factors at 20.82%, whereas algorithms were explained by only 12.10%. Perseverance was the CT approach with the highest variance explained at 20.94%, whereas creating had the lowest of the CT approaches at 11.38%.

Model 3 $(CTMIN_{ij} = \gamma_{00} + \gamma_{01} * RPPNET_j + \gamma_{02} * POVERTY_j + \gamma_{10} * EDUTIME_{ij} + \gamma_{20} * GRADELEV_{ij} + \gamma_{30} * CONFID_j + u_{0j} + r_{ij})$

Tables 2–10 were run with two level-2 predictor variables: using an RPP curriculum and adding a proxy for socioeconomic status using the percent of families with income below the poverty level. When this was added to the model there were no

significant differences between the districts for all CT concepts and approaches. The amount of variance explained by including the two variables had a broad range for the CT concepts from 17.17% (decomposition) to 61.19 % (algorithms). The CT approaches variance was explained by a broad range from 17.72% (tinkering) to 70.72% (debugging).

Model 4: $(CTMIN_{ij} = \gamma_{00} + \gamma_{01} * RPP_j + \gamma_{10} * EDUTIME_{ij} + \gamma_{20} * GRADELEV_{ij} + \gamma_{30} * CONFID_{ij} + u_{0j} + r_{ij})$

Tables 2-10 were run with only one level-2 predictor added because it was suspected there was a lot of shared variance when using the two predictor variables. The one binary level-2 predictor used in this model included using or not using the RPP curriculum with the teacher-level predictors of experience, grade level, and confidence. This predictor was of particular interest because it is a variable that can be affected by policy changes, whereas poverty level is a contextual variable that is rooted in more complex systems. Using the RPP curriculum was found to have significant effects on algorithms, debugging, and perseverance, after controlling for the effects of teacher-level predictors. It was not found to be a significant predictor for decomposition, abstraction, recognizing patterns, tinkering, creating, and collaborating. The percentage of variance explained for algorithms by just adding the RPP curriculum variable was 40.39%, for debugging was 46.90%, and for perseverance was 45.31%.

Results showed that the teacher-level variables play a significant role in determining the time spent teaching computational thinking skills. The bottom portion of Tables 2–10 shows the percentage of variance explained by the different models. The

teacher's years of experience, grade level being taught, and confidence levels for all the concepts and approaches explain 11–21% of the variance between level 1 (teachers). The engagement in an RPP curriculum and being in a district with a percentage of families that fall below the poverty level for algorithms, debugging, and perseverance explain 61–71% (Model 3) of the variance between level 2 (districts). The engagement in an RPP explains 40–47% (Model 4) of these CT concepts and approaches. In addition, 17–45% (Model 3) and 11–31% (Model 4) of the variance between districts can be explained by the variables mentioned previously for the rest of the CT concepts and approaches.

Table 2
HLM Models for Algorithms

Algorithms												
Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	75.76	7.59	9.98**	75.68	7.49	10.10**	69.11	7.75	8.91**	65.09	8.04	8.10**
RPP curriculum (γ_{01})							24.16	14.10	1.713	32.96	14.05	2.34*
Poverty(γ_{02})							-1.07	0.65	-1.63			
Teaching Experience (γ_{10})				5.87	3.98	1.47	5.87	3.98	1.48	5.87	3.97	1.48
Grade Level (γ_{20})				10.97	3.23	3.40**	10.97	3.22	3.40**	10.97	3.22	3.41**
Confidence (γ_{30})				17.16	7.97	2.15*	17.16	7.96	2.16*	17.16	7.95	2.16*
Random Effect	Variance		χ^2_{27}	Variance		χ^2_{31}	Variance		χ^2_{15}	Variance		χ^2_{20}
Level 1 (r)	3895.40			3423.94			3414.22			3404.40		
Level 2 (u_0)	390.05			417.19			161.93			248.67		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	90.90%			12.10%			12.35%			12.60%		
Level 2	9.10%						61.19%			40.39%		
Deviance (df)	1635.21 (2)			1598.09 (2)			1582.61 (2)			1587.87 (2)		

Table 3
HLM Models for Decomposition

Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	90.12	8.61	10.46**	89.72	8.45	10.61**	81.79	10.15	8.05**	79.82	9.66	8.26**
RPP curriculum (γ_{01})							25.78	18.37	1.40	30.96	16.82	1.84
Poverty(γ_{02})							-0.66	0.87	-0.76			
Teaching Experience (γ_{10})				7.83	4.25	1.84	7.83	4.25	1.84	7.83	4.25	1.84
Grade Level (γ_{20})				12.55	3.45	3.64**	12.55	3.45	3.64**	12.55	3.44	3.64**
Confidence (γ_{30})				28.78	8.52	3.38**	28.78	8.50	3.38**	28.78	8.50	3.39**
Random Effect	Variance	χ^2_{28}		Variance	χ^2_{35}		Variance	χ^2_{26}		Variance	χ^2_{27}	
Level 1 (r)	4784.91			3909.95			3897.74			3893.70		
Level 2 (u_0)	524.99			578.11			478.82			457.15		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	90.11%			18.29%			18.54%			18.63%		
Level 2	9.89%						17.17%			20.92%		
Deviance (df)	1665.83 (2)			1618.46 (2)			1605.62 (2)			1609.58 (2)		

Table 4

HLM Models for Patterns

Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	81.20	8.01	10.14**	81.05	7.88	10.28**	74.56	9.22	8.09**	71.36	9.00	7.93**
RPP curriculum (γ_{01})							22.10	16.69	1.32	29.88	15.70	1.90
Poverty(γ_{02})							-0.96	0.78	-1.22			
Teaching Experience (γ_{10})				10.37	4.19	2.48*	10.37	4.18	2.49*	10.37	4.17	2.49*
Grade Level (γ_{20})				8.88	3.40	2.61**	8.88	3.38	2.62*	8.88	3.39	2.62**
Confidence (γ_{30})				29.43	8.38	3.51**	29.43	8.35	3.52**	29.43	8.35	3.52**
Random Effect	Variance	χ^2_{26}		Variance	χ^2_{31}		Variance	χ^2_{21}		Variance	χ^2_{24}	
Level 1 (r)	4518.75			3785.74			3760.20			3762.63		
Level 2 (u_0)	415.77			462.79			335.35			359.51		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	91.58%			16.22%			16.79%			16.73%		
Level 2	8.42%						27.54%			22.32 %		
Deviance (df)	1656.37 (2)			1612.48 (2)			1598.83 (2)			1603.49 (2)		

Table 5

HLM Models for Abstraction

Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	75.16	7.80	9.64**	74.83	7.64	9.79	71.16	8.76	8.13**	66.65	8.95	7.45**
RPP curriculum (γ_{01})							15.19	15.89	0.96	25.33	15.63	1.62
Poverty(γ_{02})							-1.20	0.74	-1.62			
Teaching Experience (γ_{10})				9.87	4.28	2.31*	9.87	4.27	2.31*	9.87	4.27	2.31*
Grade Level (γ_{20})				13.77	3.47	3.97**	13.77	3.46	3.98**	13.77	3.46	3.98**
Confidence (γ_{30})				29.19	8.57	3.41**	29.19	8.55	3.42**	29.19	8.55	3.41**
Random Effect	Variance	χ^2_{22}		Variance	χ^2_{27}		Variance	χ^2_{17}		Variance	χ^2_{22}	
Level 1 (r)	4996.69			3956.27			3938.95			3939.53		
Level 2 (u_0)	325.58			394.57			247.04			334.47		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	93.88%			20.82%			21.17%			21.16%		
Level 2	6.12%						37.39%			15.23%		
Deviance (df)	1669.16 (2)			1617.44 (2)			1603.79 (2)			1609.37 (2)		

Table 6

HLM Models for Tinkering

Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	89.92	7.73	11.63**	89.67	7.69	11.67**	83.44	9.34	8.94**	80.74	8.87	9.10**
RPP curriculum (γ_{01})							1.26	16.93	1.26	27.89	15.51	1.80
Poverty(γ_{02})							-.82	0.79	-1.03			
Teaching Experience (γ_{10})				5.36	4.42	1.21	5.36	4.40	1.22	5.36	4.40	1.22
Grade Level (γ_{20})				12.64	3.58	3.53**	12.64	3.57	3.54**	12.64	3.57	3.54**
Confidence (γ_{30})				23.53	8.84	2.66**	23.53	8.80	2.67**	25.53	8.81	2.67**
Random Effect	Variance	χ^2_{22}		Variance	χ^2_{26}		Variance	χ^2_{19}		Variance	χ^2_{21}	
Level 1 (r)	4901.09			4210.82			4177.74			4185.22		
Level 2 (u_0)	321.83			378.42			311.35			299.26		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	93.95%			19.38%			14.76%			14.61%		
Level 2	6.05%						17.72%			20.92%		
Deviance (df)	1666.38 (2)			1625.72 (2)			1612.82 (2)			1617.09 (2)		

Table 7

HLM Models for Creating

Fixed Effect	Coeff.	Model 1			Model 2			Model 3			Model 4		
		S.E.	t-ratio		Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	75.88	7.23	10.49**		75.73	7.16	10.58**	74.18	8.11	9.14**	69.48	8.50	8.17**
RPP curriculum (γ_{01})								8.94	14.73	0.61	19.29	14.83	1.30
Poverty(γ_{02})								-1.27	0.69	-1.85			
Teaching Experience (γ_{10})					7.52	4.00	1.88	7.52	3.99	1.88	7.52	4.00	1.88
Grade Level (γ_{20})					7.80	3.24	2.40**	7.80	3.24	2.41*	7.80	3.24	2.40*
Confidence (γ_{30})					22.83	8.00	2.85**	22.83	7.99	2.86	22.83	8.00	2.85**
Random Effect	Variance		χ^2_{24}		Variance		χ^2_{27}	Variance		χ^2_{17}	Variance		χ^2_{23}
Level 1 (r)	3892.84				3449.70			3445.58			3450.14		
Level 2 (u_0)	319.19				348.03			205.89			311.05		
Variance	Partitioned				Explained			Explained			Explained		
Level 1	92.42%				11.38%			11.49%			11.37%		
Level 2	7.58%							40.84%			10.63%		
Deviance (df)	1633.93 (2)				1597.92 (2)			1584.74 (2)			1590.85 (2)		

Table 8

HLM Models for Debugging

Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	74.45	7.55	9.86**	74.26	7.40	10.03**	68.42	7.75	8.83**	63.83	7.94	8.04**
RPP curriculum (γ_{01})							23.49	14.17	1.66	33.10	13.97	2.37*
Poverty(γ_{02})							-1.07	0.65	-1.66			
Teaching Experience (γ_{10})				9.41	4.39	2.14*	9.41	4.37	2.15*	9.41	4.37	2.15*
Grade Level (γ_{20})				13.90	3.56	3.90**	13.90	3.55	3.92**	13.90	3.55	3.92**
Confidence (γ_{30})				27.70	8.79	3.15**	27.70	8.75	3.17*	27.70	8.75	3.17**
Random Effect	Variance		χ^2_{19}	Variance		χ^2_{24}	Variance		χ^2_{12}	Variance		χ^2_{16}
Level 1 (r)	5148.82			4162.14			4128.70			4126.19		
Level 2 (u_0)	263.49			325.08			95.17			172.61		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	95.13%			19.16%			19.81%			19.86%		
Level 2	4.87%						70.72%			46.90%		
Deviance (df)	1672.42 (2)			1623.29 (2)			1607.51 (2)			1612.88 (2)		

Table 9

HLM Models for Perseverance

Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	100.42	8.93	11.24**	100.26	8.87	11.30**	91.72	9.04	10.15**	86.41	9.29	9.30**
RPP curriculum (γ_{01})							31.00	16.44	1.89	42.99	16.26	2.64*
Poverty(γ_{02})							-1.36	0.76	-1.78			
Teaching Experience (γ_{10})				7.25	4.66	1.56	7.25	4.62	1.57	7.25	4.63	1.57
Grade Level (γ_{20})				15.44	3.78	4.09**	15.44	3.75	4.12**	15.44	3.75	4.11**
Confidence (γ_{30})				33.79	9.32	3.63**	33.79	9.25	3.65**	33.79	9.26	3.65**
Random Effect	Variance		χ^2_{27}	Variance		χ^2_{32}	Variance		χ^2_{16}	Variance		χ^2_{21}
Level 1 (r)	5920.08			4680.57			4611.21			4624.78		
Level 2 (u_0)	488.93			596.24			222.84			326.08		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	92.37%			20.94%			22.11%			21.88%		
Level 2	7.63%						62.63%			45.31%		
Deviance (df)	1695.18 (2)			1643.10 (2)			1625.04 (2)			1631.21 (2)		

Table 10

HLM Models for Collaboration

Fixed Effect	Model 1			Model 2			Model 3			Model 4		
	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio	Coeff.	S.E.	t-ratio
Intercept (γ_{00})	108.36	8.60	12.60**	108.07	8.42	12.83**	101.50	9.36	10.84**	97.07	9.37	10.36**
RPP curriculum (γ_{01})							24.55	17.00	1.44	34.56	16.39	2.11
Poverty(γ_{02})							-1.19	0.79	-1.51			
Teaching Experience (γ_{10})				9.27	4.63	2.00*	9.27	4.61	2.01*	9.27	4.61	2.01*
Grade Level (γ_{20})				15.31	3.75	4.08**	15.31	3.74	4.09**	15.31	3.74	4.09**
Confidence (γ_{30})				27.37	9.27	2.95**	27.37	9.23	2.97*	27.37	9.23	2.96**
Random Effect	Variance		χ^2_{23}	Variance		χ^2_{29}	Variance		χ^2_{17}	Variance		χ^2_{21}
Level 1 (r)	5711.10			4628.21			4590.10			4596.16		
Level 2 (u_0)	430.62			496.98			273.66			341.24		
Variance	Partitioned			Explained			Explained			Explained		
Level 1	92.99%			18.96%			19.63%			19.52%		
Level 2	7.01%						44.94 %			31.34%		
Deviance (df)	1689.42 (2)			1640.34 (2)			1625.16 (2)			1630.57 (2)		

Discussion

Teaching CT to students at the elementary level is necessary for developing problem-solving skills in our changing world, but many competing priorities make implementation difficult. Integrating CT into current science practices is one strategy gaining attention by researchers trying to accommodate educators' packed teaching schedules. The framework Waterman et al. (2020) used—Exist, Enhance, and Extend—is a strategy that may be beneficial for implementation efforts. To implement this type of framework, it is essential to figure out the baseline CT usage before a more intentional integration is executed. In addition, examining different variables such as teacher and district qualities to see if there is a relationship with the frequency of CT concepts and approaches being taught can inform what parts of the system can be leveraged to impact reform efforts.

According to Bronfenbrenner's Ecological Systems Theory (1979), different layers or systems in a child's environment influence their development and learning. These environments are nested within one another in different layers that interact directly and indirectly. Varied factors or layers can have an impact on a child's learning of CT such as teachers and district initiatives. If certain variables are determined to affect CT usage positively, teachers and district leaders can implement these variables to improve teaching and learning in their schools. This study was performed to determine the landscape of what already exists in classrooms throughout one state and analyze if certain variables are more conducive to incorporating CT within the school day.

Teacher factors and district factors were analyzed using HLM to determine if they had an impact on the teaching and learning of computational thinking skills. Three

teacher level variables were found to have a significant relationship with increasing the frequency of CT concepts and approaches which included grade level taught, teachers' confidence levels, and teacher. This study sought to answer three research questions by analyzing data collected from teachers in a cross-sectional survey about CT integration in science.

In answering the first research question concerning the frequency of CT concepts and approaches taught in K–5 classrooms, the data analysis indicated that certain concepts and approaches are more readily integrated into science instruction, such as decomposition, collaboration, persevering, and tinkering. Future implementation efforts may focus on enhancing these areas of CT during science education and on educating science teachers about the role of creating, debugging, patterns, and abstraction in science while taking the CT-TPACK framework into consideration. Recognizing patterns is a concept that science teachers with an aligned curriculum to the NGSS should prioritize, not only because patterns are one of the engineering practices but are also one of the NGSS cross-cutting concepts (NGSS Lead States, 2013). The concept of recognizing patterns may require teachers receive professional development on how it explicitly fits into the curriculum and where there are opportunities for teaching it to their students.

In answering the second research question, concerning the analysis of teacher characteristics in relation to the frequency of CT concepts and approaches, the data show that experience teaching, grade level, and confidence were significant variables that predicted outcomes, whereas professional development experience and levels of concern were not significant predictors. Teachers with more years of experience are more likely to engage in science activities that include CT approaches and skills of debugging,

collaboration, patterns, and abstraction. The concepts of recognizing patterns and abstraction involve utilizing higher order thinking skills when taught. Perhaps experienced teachers are more intentional in promoting higher-order thinking, which successfully advances critical thinking (Miri et al., 2007).

As the grade level increase, the frequency of teaching CT concepts and approaches increases. Developmentally, students can engage in more complex thinking as they progress through the grades that teachers consider when teaching their lessons. Rijke et al. (2018) found that students between the ages of 6 and 12 increased their usage of performing abstraction on a task as they increased in age. Kale et al.'s (2018) study looking at teachers' CT skills and usage found primary school teachers used significantly fewer CT skills than teachers from the secondary level. Teachers' level of confidence in teaching CT was also a significant variable for determining CT frequency. When a teacher feels like they can successfully achieve results, they are more apt to conduct the practice. Confidence in CT increases when teachers gain experience with explicit CT curriculum (Chalmers, 2018). In addition, professional development opportunities have been shown to improve teachers' confidence and self-efficacy when integrating CT (Dong et al., 2019; Jaipal-Jamani & Angeli, 2017; Zhao et al., 2020). Students will not understand CT until teachers understand and feel confident about their abilities (Israel et al., 2015).

While professional development is commonly strongly linked to implementation (Penuel et al., 2007), most teachers had “no to little” professional development in computer science implementation, and only 30% had more than three hours. The overall lack of professional development experience, and the idea that those with professional

development might have more critically judged their science activities for CT, might explain these outcomes. A similar pattern was seen in the teachers' level of concern for implementing CT in that they had low stages of concern. While research has indicated that as teachers increase their concern for innovation, they engage in the innovation more often (Hall & Hord, 2006; Hall et al., 2011). Most teachers (61%) were only at the beginning stages of concern for implementing CT meaning they may have heard something about it or want to know more about it, but other responsibilities take priority and they have not engaged in the innovation.

In answering the third research question, concerning the analysis of district-level variables in relation to the frequency of CT concepts and approaches, the data show that school classification (urban, urban ring, rural, or suburban) and the combined factors of SES and participation in an RPP did not predict the use of CT in the science curriculum. However, when isolating each predictor variable, there were significant differences. When using only poverty as the predictor variable, there were differences in the teaching and learning of algorithms, abstraction, creating, debugging, perseverance, and collaboration. As the percentage of families below the poverty level increased, the use of CT concepts and approaches decreased. There were also differences when using only the RPP curriculum as the predictor variable for algorithms, debugging, and perseverance. The interaction between SES levels and participation in the RPP was less significant due to the shared variance created by the RPP curriculum districts being more affluent when using only the 15 districts through the HLM analysis. While there is some indication that the RPP influences the use of CT integration in science, further studies involving more

participation by less affluent districts in the RPP are warranted to accurately determine if an RPP predicts more CT usage for all concepts and approaches.

This research study has demonstrated that districts using the RPP curriculum may be more apt to integrate CT into their everyday lessons. At the time of this study, the RPP was not intentionally supporting CT through curriculum or professional learning experiences. Teachers who were part of the RPP received ongoing mandatory science professional development focused on NGSS aligned science instruction where CT is a small part of these standards (GEMS-Net, 2020). This reasoning may have contributed to the higher frequency of CT concepts and approaches in districts that were part of the RPP. In a recent case study performed by Ketelhut et al. (2019), teachers found when they had opportunities for time, space, and support to discuss their science lessons where they integrated CT, they had deepened science learning. Being part of an RPP and having mandatory PD allows teachers to have this structure in place where they are given the time and space to discuss what is going well in their instruction.

Limitations

One potential challenge with the study is that the data collected is based on teacher self-reported recollections and estimations of the time they spend on lessons that incorporate CT concepts and approaches. Although the survey was anonymous, teachers may still feel compelled to overestimate their time spent on CT skills. Determining the frequency of CT is also based on the teacher's perspectives of what they are doing in the classroom. However, a detailed description of each concept and approach using definitions, pictures, and examples is included in the survey to give teachers the information needed to answer the question more precisely. A sampling limitation likely

occurred because teachers who participated in the survey may have a higher level of interest and motivation around the subject matter than those who did not participate. Some participants were asked to take the survey during school faculty meetings per principal or superintendent recommendation. Another limitation included using data from only 15 out of 30 districts that participated in the survey due to some districts having fewer than five teachers participating. Although these data requirements provided more reliable estimates in the HLM analysis, preliminary analysis shows that the excluded districts were mostly those serving lower SES communities. Increased representation from the districts not included in the analysis along with more participation from less affluent districts that are part of the RPP would reduce any biased estimates and give a more robust analysis.

Implications and recommendations

This research brings awareness to the complex systems associated with elementary school instruction that impact the implementation of new innovations. Based on the data from teachers' surveys, it is clear that there is great potential to integrate CT into daily science instruction. While the concepts and approaches of decomposition, collaboration, persevering, and tinkering might integrate more effortlessly, creating, debugging, patterns and abstraction may need extra support through professional development and targeted curricula. In addition, it is clear that CT is integrated into science lessons more as grade levels progress. While this may be developmentally appropriate, it may also be a function of extra support needed at the earliest grades, particularly as the concern for the digitally literate problem-solvers increase to meet the needs of our complex world. The increase of teachers' years of experience with teaching

and their confidence in teaching CT predicted a greater amount of CT integration. Therefore, additional support for novice teachers and professional development to increase self-efficacies for all teachers is recommended. In addition, pairing experienced, confident teachers with less experienced and less confident teachers may increase CT integration. While teacher-level factors had a greater influence on the integration of CT than district level, we learned that participation in an RPP may positively influence CT integration and that poverty may have a negative influence. Providing more resources at all levels of the educational system, but particularly for schools in low-income areas is needed to ensure children develop computational thinking skills effectively and equitably so they can be active participants in a digitally rich and problem-driven society. Additional research is needed on how the systems approach and sustainable teacher development of an RPP may leverage science curricula and increase science teachers' ability to integrate computational thinking into their instruction.

References

- 2018 State of Computer Science Education. (2018). Retrieved from <https://advocacy.code.org/>
- Adler, R. F., & Kim, H. (2018). Enhancing future K-8 teachers' computational thinking skills through modeling and simulations. *Education and Information Technologies*, 23(4), 1501-1514. <https://doi.org/10.1007/s10639-017-9675-1>
- Angeli, C., & Jaipal-Jamani, K. (2018). Preparing pre-service teachers to promote computational thinking in school classrooms. In *Computational thinking in the stem disciplines: Foundations and research highlights* (pp. 127-150). Springer, International Publishing.
- 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. *Journal of Educational Technology & Society*, 19(3), 47-57.
- Baek, Y., Wang, S., Yang, D., Ching, Y. H., Swanson, S., & Chittoori, B. (2019). Revisiting second graders' robotics with an Understand/Use-Modify-Create (U2MC) strategy. *European Journal of STEM Education*, 4(1), 1-12. <https://doi.org/10.20897/ejsteme/5772>
- Bandura, A. (1986). Social foundations of thought and action: A social cognitive theory. (pp. 5-107). Prentice Hall.
- Banilower, E. R., Smith, P. S., Malzahn, K. A., Plumley, C. L., Gordon, E. M., & Hayes, M. L. (2018). Report of the 2018 NSSME+. Chapel Hill, NC: Horizon Research, Inc.
- Barefoot Computing. (n.d.). Retrieved from <https://www.barefootcomputing.org>
- 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/10.1145/2676723.2693616>
- Bart, A. C. (2015, February). Situating computational thinking with big data: Pedagogy and technology. In *Proceedings of the 46th ACM technical symposium on computer science education* (pp. 719-719).
- Berry, M. (2013). *Computing in the National Curriculum: A Guide for Primary Teachers*, Computing at School, Bedford.
- Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in the early childhood classroom*. Routledge.
- Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum.

Computers & Education, 72, 145-157.
<https://doi.org/10.1016/j.compedu.2013.10.020>

- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education-Implications for policy and practice* (No. JRC104188). Joint Research Centre (Seville site).
- Boulden, D. C., Wiebe, E., Akram, B., Aksit, O., Buffum, P. S., Mott, B., Boyer, K. E., & Lester, J. (2018). Computational thinking integration into middle grades science classrooms: Strategies for meeting the challenges. *Middle Grades Review*, 4(3) 1-17. <https://scholarworks.uvm.edu/mgreview/vol4/iss3/5>
- Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017, November). Development of computational thinking skills through unplugged activities in primary school. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 65-72). <https://doi.org.uri.idm.oclc.org/10.1145/3137065.3137069>
- Bronfenbrenner, U. (1996). *The ecology of human development experiments by nature and design*. Harvard University Press.
- Bureau of Labor Statistics (2021). U.S. Department of Labor, Computer and Information Research Scientists: *Occupational Outlook Handbook*. Retrieved from <https://www.bls.gov/ooh/computer-and-information-technology/computer-and-information-research-scientists.htm>
- Butler, D., & Leahy, M. (2021). Developing preservice teachers' understanding of computational thinking: A constructionist approach. *British Journal of Educational Technology*, 52(3), 1060–1077. <https://doi.org/10.1111/bjet.13090>
- Chalmers, C. (2018). Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17, 93-100. <https://doi.org/10.1016/j.ijcci.2018.06.005>
- Coburn, C. E., & Penuel, W. R. (2016). Research–practice partnerships in education: Outcomes, dynamics, and open questions. *Educational Researcher*, 45(1), 48-54. <https://doi.org/10.3102/0013189X16631750>
- Coburn, C. E., Penuel, W. R., & Farrell, C. C. (2021). Fostering educational improvement with research-practice partnerships. *Phi Delta Kappan*, 102(7), 14-19. <https://doi.org/10.1177/00317217211007332>
- Computational Thinking Curriculum, Computer Science Lessons. (n.d.). Retrieved from <https://nicerc.org/curricula/computer-science/>
- Computational Thinking for Educators—Unit 1 - Introducing Computational Thinking. (2015). Retrieved from <https://computationalthinkingcourse.withgoogle.com/unit>

- COUNT, R. I. (2018). Policy & Advocacy for Rhode Island's Children. Retrieved from <http://www.rikidscount.org/>
- Creswell, J. W. (2014). The selection of a research approach. *Research design: Qualitative, quantitative, and mixed methods approaches*, 3-24.
- Denning, P. J., & Tedre, M. (2019). *Computational Thinking*. MIT Press.
- Dillman, D. A., Smyth, J. D., & Christian, L. M. (2014). *Internet, phone, mail, and mixed-mode surveys: The tailored design method*. John Wiley & Sons.
- Dong, Y., Catete, V., Jocius, R., Lytle, N., Barnes, T., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2019, February). PRADA: A practical model for integrating Computational thinking in K–12 education. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 906-912). <https://doi.org/10.1145/3287324.3287431>
- GEMS-Net (2020). GEMSNET Guiding Education in Math and Science Network. Retrieved from <https://web.uri.edu/gemsnet/>
- Google Inc. (2015). Searching for Computer Science: Access and Barriers in U.S. K–12 Education. Retrieved from https://services.google.com/fh/files/misc/searching-for-computer-science_report.pdf
- Google Inc. & Gallup Inc. (2017, December). Encouraging Students Toward Computer Science Learning. Results From the 2015-2016 Google-Gallup Study of Computer Science in U.S. K–12 Schools (Issue Brief No. 5). Retrieved from <https://goo.gl/iM5g3A>.
- Davis, E. & Smithey, J. (2009). Beginning teachers moving toward effective elementary science teaching. *Science Education (Salem, Mass.)*, 93(4), 745–770. <https://doi.org/10.1002/sce.20311>
- Grover, S. (2018). The 5th ‘C’ of 21st century skills. *Try computational thinking (not coding)*. (March 13). Retrieved from EdSurge News: <https://www.edsurge.com/news/2018-02-25-the-5th-c-of-21st-century-skills-try-computational-thinking-not-coding>.
- Guba, E. G. & Lincoln, Y.S. (1994). Competing paradigms in qualitative research. In N. Denzin & Y. S. Lincoln, *Handbook of qualitative research* (1st Ed.). (pp.105-117). Sage Publications, Inc.
- Hall, G. E., & Hord, S. M. (2006). *Implementing change: Patterns, principles, and potholes*. Pearson.
- Hall, G. E., Hord, S. M., Aguilera, R., Zepeda, O., & von Frank, V. (2011). Implementation: Learning builds the bridge between research and practice. *The Learning Professional*, 32(4), 52-57.

- Hestness, E., Ketelhut, D. J., McGinnis, J. R., & Plane, J. (2018). Professional knowledge building within an elementary teacher professional development experience on computational thinking in science education. *Journal of Technology and Teacher Education*, 26(3), 411-435. <https://www.learntechlib.org/primary/p/181431/>
- Hord, S. M., & Roussin, J. L. (2013). *Implementing change through learning: Concerns-based concepts, tools, and strategies for guiding change*. Corwin Press.
- Hsu, Y. S. (2015). The development of teachers' professional learning and knowledge. *Development of science teachers' TPACK*, 3-15.
- Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, 82, 263-279. <https://doi.org/10.1016/j.compedu.2014.11.022>
- Iyer, S. (2019). Teaching-Learning of Computational Thinking in K–12 Schools in India. In *Computational Thinking Education* (pp. 363-382). Springer.
- ISTE, C. (2011). Computational Thinking in K–12 Education leadership toolkit. *Computer Science Teacher Association*: http://csta.acm.org/Curriculum/sub/CurrFiles/471.11_CTLLeadershipToolkit-SP-vF.pdf adresinden alındı.
- Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, 26(2), 175–192. <https://doi-org.uri.idm.oclc.org/10.1007/s10956-016-9663-z>
- Kafura, D., Bart, A. C., & Chowdhury, B. (2018). A computational thinking course accessible to non-stem majors. *Journal of Computing Sciences in Colleges*, 34(2), 157-163.
- Kale, U., Akcaoglu, M., Cullen, T., & Goh, D. (2018). Contextual factors influencing access to teaching computational thinking. *Computers in the Schools*, 35(2), 69-87. <https://doi-org.uri.idm.oclc.org/10.1080/07380569.2018.1462630>
- Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 4(3), 583-596.
- Karpinski, Z., Biagi, F., & Di Pietro, G. (2021). Computational Thinking, Socioeconomic Gaps, and Policy Implications. IEA Compass: Briefs in Education. Number 12. *International Association for the Evaluation of Educational Achievement*.
- Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2019). Teacher change following a professional development experience in integrating

- computational thinking into elementary science. *Journal of Science Education and Technology*, 1-15. <https://doi.org/10.1007/s10956-019-09798-4>
- Khine, M. S. (Ed.). (2018). *Computational Thinking in the STEM Disciplines: Foundations and Research Highlights*. Springer.
- Koehler, M., & Mishra, P. (2009). What is technological pedagogical content knowledge (TPACK)? *Contemporary Issues in Technology and Teacher Education*, 9(1), 60-70.
- Laerd Statistics (2018). Spearman's correlation using SPSS Statistics. *Statistical tutorials and software guides*. Retrieved from <https://statistics.laerd.com/>
- Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2019). Computational thinking from a disciplinary perspective: Integrating computational thinking in K–12 Science, Technology, Engineering, and Mathematics education. *Journal of Science Education and Technology*, 29(1), 1-8. (2020) 29:1–8. <https://doi.org/10.1007/s10956-019-09803-w>
- Maeng, Whitworth, B. A., Bell, R. L., & Sterling, D. R. (2020). The effect of professional development on elementary science teachers' understanding, confidence, and classroom implementation of reform-based science instruction. *Science Education (Salem, Mass.)*, 104(2), 326–353. <https://doi.org/10.1002/sce.21562>
- Maltese, A. V., & Tai, R. H. (2010). Eyeballs in the fridge: Sources of early interest in science. *International Journal of Science Education*, 32(5), 669-685. <https://doi-org.uri.idm.oclc.org/10.1080/09500690902792385>
- Mattera, S., & Morris, P. (2017). Counting on Early Math Skills: Preliminary Kindergarten Impacts of the Making Pre-K Count and High 5s Programs. MDRC.
- Mcintosh, K., Horner, R., Chard, D., Boland, J., & Good, R. (2006). The use of reading and behavior screening measures to predict nonresponse to school-wide positive behavior support: A longitudinal analysis. *School Psychology Review*, 35(2), 275-291. <https://doiorg.uri.idm.oclc.org/10.1080/02796015.2006.12087992>
- Metz, S. S. (2007). Attracting the engineers of 2020 today. *Women and minorities in science, technology, engineering, and mathematics: Upping the numbers*, 184-209.
- Miri, B., David, B. C., & Uri, Z. (2007). Purposely teaching for the promotion of higher-order thinking skills: A case of critical thinking. *Research in science education*, 37(4), 353-369. <https://doi.org/10.1007/s11165-006-9029-2>
- Mulvey, K. L., McGuire, L., Hoffman, A. J., Hartstone-Rose, A., Winterbottom, M., Balkwill, F., Fields, G. E., Burns, K., Drews, M., Chatton, M., Eaves, N., Law, F., Joy, A., & Rutland, A. (2020). Learning hand in hand: Engaging in research–

- practice partnerships to advance developmental science. *New Directions for Child and Adolescent Development*, 2020 (172), 125-134. <https://doi-org.uri.idm.oclc.org/10.1002/cad.20364>
- National Center for Education Statistics (NCES) U.S. Department of Education. (n.d.). Retrieved from <https://nces.ed.gov/>
- National Research Council. (2012). *A framework for K–12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press.
- NGSS Lead States. (2013). *Next generation science standards: For states, by states*. Washington, D.C.: National Academies Press.
- Noonan, R., & United States. Economics Statistics Administration, issuing body. (2017). *STEM jobs: 2017 update* (ESA issue brief, 17-02). Washington, DC: U.S. Department of Commerce, Economics and Statistics Administration, Office of the Chief Economist.
- Ouyang, Y., Hayden, K. L., & Remold, J. (2018, February). Introducing Computational Thinking through Non-Programming Science Activities. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 308-313). ACM. <https://doi.org/10.1145/3159450.3159520>
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc. Peel, A., Sadler, T. D., & Friedrichsen, P. (2019). Learning natural selection through computational thinking: Unplugged design of algorithmic explanations. *Journal of Research in Science Teaching*, 56(7), 983-1007. <https://doi.org/10.1002/tea.21545>
- Penuel, W. R., Coburn, C. E., & Gallagher, D. J. (2013). Negotiating problems of practice in research–practice design partnerships. *National Society for the Study of Education Yearbook*, 112(2), 237-255.
- Penuel, W. R., Fishman, B. J., Yamaguchi, R., & Gallagher, L. P. (2007). What makes professional development effective? Strategies that foster curriculum implementation. *American Educational Research Journal*, 44(4), 921-958. <https://doi.org/10.3102/0002831207308221>
- Plumley, C. L. (2019). 2018 NSSME+: Status of elementary school science. Chapel Hill, NC: Horizon Research, Inc.
- Prottsman, K. (2019). *Computational Thinking Meets Student Learning: Extending the ISTE standards*. International Society for Technology in Education.
- Raudenbush, S. W., & Bryk, A. S. (2002). *Hierarchical linear models: Applications and data analysis methods* (2nd Edition). Sage Publications, Inc.

- Rich, P. J., Jones, B., Belikov, O., Yoshikawa, E., & Perkins, M. (2017). Computing and engineering in elementary school: The effect of year-long training on elementary teacher self-efficacy and beliefs about teaching computing and engineering. *International Journal of Computer Science Education in Schools*, 1(1), 1-20. <https://doi.org/10.21585/ijcses.v1i1.6>
- Rijke, W. J., Bollen, L., Eysink, T. H., & Tolboom, J. L. (2018). Computational thinking in primary school: An examination of abstraction and decomposition in different age groups. *Informatics in Education*, 17(1), 77.
- Rischar, J. F. (2007). *High noon: 20 global problems, 20 years to solve them*. Hachette UK.
- Ruberg, L. F., & Owens, A. (2017). A future-focused education: Designed to create the innovators of tomorrow. In *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 367-392). Springer, Cham.
- Sandholtz, J. H., & Ringstaff, C. (2020). Offering modest supports to extend professional development outcomes and enhance elementary science teaching. *Professional Development in Education*, 1-16. <https://doi.org/10.1080/19415257.2020.1725594>
- Sands, P., Yadav, A., & Good, J. (2018). Computational thinking in K–12: In-service teacher perceptions of computational thinking. In *Computational thinking in the STEM disciplines* (pp. 151-164). Springer, Cham.
- Sengupta, P., Kinnebrew, J., Basu, S., Biswas, S., & Clark, G. (2013). Integrating Computational thinking with K–12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18(2), 351-380. <https://doi.org/10.1007/s10639-012-9240-x>
- Sweetman, S. (Principal Investigator). (2018-2020). *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (Award number: 1813224) [Grant] National Science Foundation.
- Teachers & Administrators. (n.d.). Retrieved from <https://www.ride.ri.gov/TeachersAdministrators/EducatorCertification.aspx>.
- Usengül, L., & Bahçeci, F. (2020). The effect of LEGO WeDo 2.0 Education on academic achievement and attitudes and computational thinking skills of learners toward science. *World Journal of Education*, 10(4), 83- 93. <https://doi.org/10.5430/wje.v10n4p83>
- 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(1), 53-64. <https://doi.org/10.1007/s10956-019-09801-y>

- Weinberg, A. E. (2013). *Computational thinking: An investigation of the existing scholarship and research* (Doctoral dissertation, Colorado State University).
- Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1), 127-147. <https://doi.org/10.1007/s10956-015-9581-5>
- White, E., & Shakibnia, A. F. (2019). State of STEM: Defining the Landscape to Determine High-Impact Pathways for the Future Workforce. In *Proceedings of the Interdisciplinary STEM Teaching and Learning Conference* (Vol. 3, No. 1, p. 4). DOI: 10.20429/stem.2019.030104
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35. <https://doi.org/10.1145/1118178.1118215>
- Woltman, H., Feldstain, A., MacKay, J. C., & Rocchi, M. (2012). An introduction to hierarchical linear modeling. *Tutorials in Quantitative Methods for Psychology*, 8(1), 52-69.
- Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In *Emerging research, practice, and policy on computational thinking* (pp. 205-220). Springer, Cham.
- Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends*, 60(6), 565-568. <https://doi.org/10.1007/s11528-016-0087-7>
- Yadav, A., Larimore, R., Rich, K., & Schwarz, C. (2019, March). Integrating computational thinking in elementary classrooms: Introducing a toolkit to support teachers. In *Society for Information Technology & Teacher Education International Conference* (pp. 347-350). Association for the Advancement of Computing in Education (AACE).
- Yadav, A., Mayfield, C., Zhou, N., Hambruch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, 14(1), 5, 1-16. <https://doi.org/10.1145/2576872>
- Zhao, M., Zhao, M., Wang, X. H., & Ma, H. L. (2020, December). Promoting Teaching Self-efficacy in Computational Thinking of School Teachers. In *2020 Ninth International Conference of Educational Innovation through Technology (EITT)* (pp. 235-239). IEEE. doi:10.1109/EITT50754.2020.00048

CHAPTER 4

TEACHER PRACTICES FOR COMPUTATIONAL THINKING IMPLEMENTATION IN ELEMENTARY SCIENCE

Currently in preparation for the *Elementary School Journal*

Department of Education, University of Rhode Island
Kingston, Rhode Island, 02881

Abstract

Computational thinking (CT) is a key component of computer science and a foundational thinking process that is essential for students to learn in K–12 classrooms. Integrating CT into science instruction provides students a more authentic experience of how scientists use integrated skills to conduct research in the field while allowing teachers to meet the multiple demands of fitting in the computing skills that children need within their school day. Although implementing CT can be a complex process, it can be easily integrated into science lessons with the right policies, programs, and practices in place. This paper explores the current practices of CT in a northeast U.S. state based on survey data collected from teachers. The research uncovers how much teaching time is spent teaching science and the percentage of lessons that have CT concepts (algorithms, decomposition, abstraction, pattern recognition) and approaches (tinkering, creating, debugging, persevering, collaborating) present in the science instruction while describing what these lessons look like. This paper then discusses the next steps for implementation efforts based on the described CT practices in elementary science.

Introduction

We live in a world where computers are ubiquitous in almost every aspect of our lives. It takes little effort to find information about scientific discoveries or data on just about any topic. This instant access to information has caused many educators to reconsider the types of knowledge our students need to participate in today's societies. The necessity of focusing on foundational computer skills and how children think is now paramount considering a large portion of today's careers require employees to have computing skills (Mills et al., 2021; Montoya, 2017; Webb et al., 2017). Introducing computer science to early learners could serve as a means for increasing diversity in this field while allowing all students access to the foundational knowledge they deserve (Mills et al., 2021).

Computer science is important for developing skills associated with problem solving, reasoning, creativity, and metacognition (Code.org et al., 2021). It is one of the fastest growing careers with employment opportunities predicted to grow 19% by 2026 (Bureau of Labor Statistics, 2021). Leaders in the field need to increase opportunities for students from urban, rural, and low socioeconomic areas to engage in computer science to decrease disparities in this vocation (Code.org et al., 2021). Furthermore, they need to adapt teaching to the cultural background of all students (Ryoo et al., 2019). Policies adopted in computer science for K–12 education should allow all students access to computer science and not exclude any populations from learning opportunities in this rapidly expanding field.

One foundational component of computer science is computational thinking (CT). This process was introduced by Jeannette Wing in 2006 in her well-known article where

she described CT as “a way of solving problems, designing systems, and understanding human behavior that draws on concepts fundamental to computer science [...] a fundamental skill for everyone, not just computer scientists” (Wing, 2006, p. 33).

Computational thinking is a thought process that utilizes different concepts and approaches to organize information efficiently. While computers have automated many of these processes for us, humans need to understand the thinking necessary to arrive at solutions for giving directions or for programming a computer to conduct the steps of their thinking. This type of thinking is a foundational skill necessary for living and working in the twenty-first century.

This process can be broken down into constructs which include CT concepts and CT approaches (Berry, 2013). The concepts, which are the main characteristics of CT and the foundation to computer science, consist of creating algorithms, using abstraction, using decomposition, and recognizing patterns (Berry, 2013; Prottzman, 2019). The approaches to learning, or the activities conducted to achieve the concepts, consist of tinkering, creating, debugging, persevering, and collaborating (Berry, 2013). Although multiple variations of the definition and the component parts of CT exist, researchers, practitioners, and policy makers all are in consensus that CT is a skill all students should have in these modern times.

Teaching CT to students at the elementary level is necessary for this changing world but there are many other priorities that make it difficult to add CT into an already packed teaching schedule. Even though the purpose of why CT should be included in the classroom is starting to be understood and policies are beginning to be put into place, teachers need sufficient support and examples of what CT looks like in instruction so

implementation can be done at a large scale in elementary classrooms. One of the best ways to implement CT is to integrate it into what is already happening in subject areas, such as science with a more focused approach (Waterman, 2019). Currently, CT has been implemented at the secondary level, but there is a growing call in the literature to introduce CT in elementary lessons so all children will have access and possibly develop an interest in computer science (Dasgupta et al., 2017; Mills et al., 2021).

The purpose of this exploratory study is to examine the current landscape of what exists in classrooms throughout a state in northeastern United States and describe science lessons that can be enhanced and extended. This analysis aims to answer the following research questions:

1. How often do K–5 teachers teach science? How long do K–5 teachers teach science lessons?
2. What percentage of time do teachers report teaching CT concepts and approaches in their science classes?
3. How accurate are CT concepts and approaches in the examples given by K–5 teachers currently taught in science classrooms?
4. How do teachers describe a lesson where they include the different CT concepts (decomposition, abstraction, algorithms, or pattern recognition) or approaches (tinkering, debugging, creating, persevering, and collaboration) in the teaching and learning process?

Theoretical Framework

Roger Bybee (2013) developed a model that incorporates four dimensions used in STEM reform —Purpose, Policy, Program, and Practice (4P’s)—that can be helpful with implementing CT in science instruction. In the 4P’s model, the *Purpose* is the goal towards which researchers, practitioners, administrators, policymakers, parents, and students strive. The goal of CT is evident in the International Society for Technology in Education’s (ISTE) 2022 goals for being a computational thinking learner which are to “set professional learning goals to explore and apply teaching strategies for integrating CT practices into learning activities in ways that enhance student learning of both the academic discipline and CS concepts.” Having this type of synergistic goal is helpful with implementation efforts from the perspectives of both the science and computer science disciplines.

The second dimension, *Policies*, is the concrete translations of the purpose. Documents such as the CS K–12 Framework, Next Generation Science Standards (NGSS), and state computer science standards give direction and guidance for CT instruction. The third dimension, *Program*, is the curriculum materials that embody the purpose and policies that are used for different grade levels. At the elementary level, the limited CT programs developed are aimed at providing the foundation for computer science knowledge that subsequent grades can build upon. Curriculum designers can follow the guidance of the ISTE standard when developing resources in which curriculum designers “learn to recognize where and how computation can be used to enrich data or content to solve discipline-specific problems and be able to connect these opportunities to foundational CT practices and CS concepts” (ISTE, 2022).

Finally, *Practices* are the processes of teaching which include the interactions between teachers and students that include the purpose, policies, and programs of the reform. *Practices* also includes changing teaching strategies and adapting materials to the unique needs of schools and students to implement the reform. School districts can achieve change when they “leverage CT and CS experts, resources and professional learning networks to continuously improve practice” (ISTE, 2022). Moreover, they can “plan collaboratively with other educators to create learning activities that cross disciplines to strengthen student understanding of CT and CS concepts and transfer application of knowledge in new contexts” (ISTE, 2022). This 4P model frames this study, which takes a bottom-up approach to implementation. Research has shown that policy is not readily transferring into practice. A deeper look at the current practices of CT may help align programs, policy, and purpose so that children benefit from CT implementation.

Purpose of integrating CT into elementary science

The desire of researchers, practitioners, and policymakers to integrate CT at the elementary level has increased in the United States (Ketelhut et al., 2019; Lee et al., 2019; Yadav et al., 2017). The rationales for exposing children at an early age to experiences involving CT-integrated science lessons are to develop positive attitudes towards the disciplines of computer science and science, to make a difference in career aspirations, and to permit more children access to foundational skills needed in this digital age (Karpinki et al., 2021; Maltese & Tai, 2010; Weintrop et al., 2016).

The 2018 International Computer and Information Literacy Study indicated that students from less advantaged backgrounds have lower levels of CT skills than those

from more advantaged backgrounds (Karpinski et al., 2021). Therefore, it is important to create equitable CT opportunities by integrating CT into core subject areas that all children take. Science is one of the core subjects in American schooling required at all levels of education. Integrating CT into science allows students to have a more authentic view of the professional work of scientists (Denning & Tedre, 2019). Science is a discipline with many real-world problems in areas such as the environment, health, and energy needs for which CT skills can be applied like in new fields such as computational biology and computational chemistry (Lee & Malyn-Smith, 2020). CT integrated in the science curriculum has been shown to lead to a deeper understanding of science, to data analysis and modeling skills, and to better prepared citizens (Barr & Stephenson, 2011; Ketelhut et al., 2019; Yadav et al., 2017). CT also improves problem-solving, reasoning, logical-mathematical skills, and academic performance (Martinez-Garcia, 2021). The early years of schooling is the best time to start laying the foundation for students to become innovative thinkers.

Hsu et al. (2019) conducted a review of global policy initiatives for the teaching and learning of CT and found eleven different rationale themes for teaching CT around the world. Of the eleven rationale themes that emerged from the review, the United States had five rationales for the purpose of CT to be taught:

to prepare citizens for future economies in which digital literacy will be a prerequisite for work; to foster problem-solving skills, especially to help solve the country's and world's problems; to enable students to become technology producers and not just technology users; to enable participation in modern life and

society; and to support digital literacy and digital competence. (Hsu et al., 2019, p. 266–267)

Policies for Integrating CT

Standards and frameworks with goals for including CT in instruction have been put into policies; however, processes for implementation at the elementary level are still in the beginning stages in many states in the United States (Code.org et al., 2021). After Wing's (2006) call to action, aspirations to increase computational thinking in K–12 education were led by the Association for Computing Machinery (ACM), Computer Science Teacher Association (CSTA), and the National Science Foundation (NSF), which led to changes nationally (Fisher, 2016). In 2016, former President Obama, in his state of the union address, set the goal of computer science for all (CS for All), resulting in research on establishing high-quality computer science learning opportunities for all students in K–12 classrooms (Code.org et al., 2021). Most of the opportunities started at the high-school level but many researchers, practitioners, and policymakers see the value in teaching it at the elementary level (Grover, 2018, Ketelhut et al., 2019; Yadav et al., 2017).

CT has been introduced into global policies in four separate ways: (1) having the subject taught as a standalone course as educators do in England, (2) embedded in subject areas such as science and math as in Japan, (3) integrated across all disciplines as a transversal skill like in Finland, and (4) a mixture of all three. In the United States, computer science policies vary from state to state (Hsu et al., 2019). Every Student Succeeds Act (ESSA), a federal policy established in 2015, explains this variation. The ESSA addresses the need to make public schools more equitable and hold all students to

high academic standards for college and career preparation but declares states and districts in charge of the decision making for rules and regulations for their educational systems. The state and districts are responsible for developing state standards, tests aligned with those standards, and their own accountability system which all vary among the fifty states concerning computer science.

In education, policy and implementation are deeply connected, so whatever policies are enacted at the state and district level are typically what get taught in schools (Fisher, 2016). For example, almost all the northeast states have adopted the Next Generation Science Standards that were developed in 2013. These standards include CT as one of the eight practices that are taught, along with asking questions and defining problems, developing models, constructing explanations, and designing solutions, planning, and carrying out investigations, analyzing and interpreting data, engaging in argument from evidence, and communicating information. The NGSS states that computational thinking “involves strategies for organizing and searching data, creating sequences of steps called algorithms, and using and developing new simulations of natural and designed systems” (NGSS Appendix F, p. 10).

Besides having standards listed within the science discipline to further computer science efforts in the K–12 classrooms, the CS for All initiative led to the northeast state in this study adopting a state-wide initiative to bring high-quality learning experiences in computer science for all students. This CS for All initiative is between the Governor’s office and the State Department of Education. The state’s goal for computer science is to broaden participation and equity, stimulate learning and curiosity, build connections

across disciplines, encourage workforce and economic development, support students and teachers, and inform with current research (CS4RI, 2022)

In 2018, computer science standards that included computational thinking were adopted for the state's K–12 education system (CS4RI, 2022). The computer standards were formed from the key ideas of the K–12 CS Framework and the Computer Science Teachers Association (CSTA), which define the computer science knowledge all students should have to meet the demands of the 21st century. The state has put supports in place to help with implementation efforts. This includes all CS content providers will align their content and professional development with the standards; school districts will utilize a template for implementation goals called SCRIPT, which stands for School CS for ALL Resource & Implementation Planning Tool; the local CSTA will provide resources and support through communities of practice, and develop materials and resources to support CS Education Standards adoption in K–12 education (K–12 Computer Science Education Standards, 2018).

At this time, school districts in the state are not mandated to adopt the CS Education Standards. However, efforts for providing CS support have been positive. In 2019 the State Department of Education adopted policy RIGL §16-22-31 which required the Commissioner of Elementary and Secondary Education and the State Department of Education to develop a statewide curriculum framework for science that allows all students access to high-quality curriculum and instruction. The framework for science aligns with the Next Generation Standards, which include CT as an engineering practice. Local education agencies (LEAs) will need to adopt a high-quality curriculum by June

30, 2025. However, LEAs should pay attention and be selective in how CT is integrated in the curriculum they choose to implement.

Programs for Integrating CT

There are many types of programs, both computer-based and unplugged, for CT instruction. Unplugged CT activities do not use a computer; teachers use curricula such as Thinkersmith, Code.org, and CS Unplugged. Some of the computer-based programs include building-block programming, tangible programming, digital game creation, and robotics. All of these approaches include computer programming as a way to show a solution to a problem. Building-block programming uses programs such as Logo or Scratch, in which the child snaps blocks together to create computer programs with a low-floor and high-ceiling approach. Tangible programming is an approach that makes programming less abstract for children by having them use physical objects that interact with a computer through curriculum such as T-Maze and Twinkle. Digital game design entails making visual representations in the creation of games through curriculums such as CSMinds Learning Suite (Brackman et al., 2017). Robotics is a hands-on learning approach that uses programs such as LEGO Mindstorms or LEGO WeDo (Angeli & Jaipal-Jamani, 2018).

Although finding the right curriculum for integrating CT in science instruction is rather new, there has been some progress and success using different programs. Incorporating CT modules, programming tools, and using models and simulations has motivated teachers to embed CT into their instruction (Yadav et al., 2014; Adler & Kim, 2018; Ruberg & Owens, 2017; Sengupta et al., 2013). However, introducing CT only

through programming or robotics will not teach the cognitive processes necessary for CT. CT is not the same as programming (Martinez-Garcia, 2021).

The distinction between computational thinking and programming is subtle; in principle computational thinking does not require programming at all, although in practice, representing a solution to a problem as a program provides a perfect way to evaluate the solution, as the computer will execute the instructions to the letter, forcing the student to refine their solution so that it is very precise. (Webb et al., 2017, p. 449-450).

Waterman et al. (2020) have produced a framework that is useful in developing curriculum materials for integrating CT into science instruction: the Exist, Enhance, Extend Framework. The three levels in the framework increase in complexity when implementing CT. The first level, referred to as “Exist,” is when teachers can recognize different CT concepts and approaches that are present in the curriculum they already have and make connections to how they relate to technology and computers. “Enhance” is when additional tasks or lessons are created that are not central to the lesson but make clear connections to computing concepts. The last level is “Extend,” which is when new lessons or sequences are created that promote CS exploration through activities like programming.

An example of how to use this framework and integrate CT into science instruction was conducted in the northeast by Waterman et al. (2019) in a unit taught to elementary students covering populations and ecosystems. This example serves as an excellent resource for teachers because it takes a lesson that teachers typically teach and goes through the process of using the Exist, Enhance, and Extend framework. Many

states have adopted CS education standards and are looking for successful integration models to use at the school and district levels to help with their implementation efforts.

Practices for Integrating CT in Science

Designing professional development to support pre-service and in-service teachers is essential in implementing CT integrated instruction (Ketelhut et al., 2019; Lee et al., 2019; Yadav et al., 2016; Yadav et al., 2017). Teachers need resources on how to effectively integrate the concepts and approaches into their content and pedagogy (Bocconi et al., 2016). There are some cases where researchers are learning strategies on how to integrate CT in science instruction with in-service teachers and through pre-service teachers' methods courses (Adler & Kim, 2018; Ketelhut et al., 2019; Rich et al., 2020; Yadav et al., 2014). CT integration can often overwhelm teachers who are not always confident, knowledgeable, or experienced with science (Ketelhut et al., 2019).

Explicit instruction on how teachers can add CT to their current instruction is necessary to put computer science policies into practice. When providing professional development, teachers have many concerns that need to be addressed. Some of these concerns include school systems placing greater instructional focus on math and reading and giving limited time for science; educators needing more teaching time for trial and error with students who struggle with higher-order thinking involved with CT; planning time to find or create CT-infused lessons; locating funds to purchase tools and programs; and providing more practical examples that use resources that are available (Ketelhut et al., 2019). Israel et al. (2015) reported other barriers to teachers, such as a lack of technology and having a lack of computing expertise. Furthermore, teachers are concerned about administrators not being supportive; teachers want them to understand

how computing impacts pedagogy, classroom management, and evaluation of state assessments.

These concerns are all valid, but progress is being made to address them. For example, professional development opportunities have been conducted that increase teachers' self-efficacy and confidence when integrating CT (Chalmers, 2018; Jaipal-Jamani & Angeli, 2017; Rich et al., 2017). Teachers have also felt supported by embedded coaching, supportive administrators, ongoing mentoring support, and having time and space to discuss lessons (Israel et al., 2015; Ouyang et al., 2018; Ketelhut et al., 2019). Also, teachers are having success in integrating CT while including all students by using the inclusive pedagogies shared in the Universal Design for Learning framework (CAST, 2018; Mills et al., 2021). It is from these pockets of progress that future implementation efforts can build.

Another way teachers have felt supported in the classroom is by belonging to a research practice partnership (Coburn et al. 2021). Research Practice Partnerships (RPP) are long-term, mutually beneficial collaborations between researchers and educators that use evidence from research to improve decision making when solving problems of practice (Coburn & Penuel, 2016). Teachers and researchers found the benefit of an RPP when integrating CT in a middle-school science classroom. It allowed researchers to try out different strategies for implementation and allowed teachers to feel more comfortable taking risks in the classroom (Boulden et al., 2018).

Despite there being pockets of research on successful strategies for CT integration in science instruction, there still is a gap in the literature pertaining to opportunities where CT should be highlighted in science instruction. Many of the current studies that

produced strategies and supports did not take into account what was currently happening in the classroom. This lack of consideration causes concern for implementation of these strategies. Implementation needs to consider what the teachers know and are doing to effectively translate policy into practice. This research aims to share what CT concepts and approaches are present in current science instruction and to serve as a baseline for future integration efforts and teacher professional development.

Methods

This study uses an embedded mixed methods design examining a portion of survey data collected from elementary teachers in a northeast state to learn what CT concepts and approaches are happening in science lessons in K–5 classrooms. A Design-Based Implementation Research (DBIR) team participated in the analysis of part of the survey for intercoder agreement. This DBIR team were part of a larger research STEM+C project entitled *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (Sweetman, 2018-2021) (1813224). The DBIR team consisted of 25 participants with various expertise and current job titles including elementary math and science coaches, teachers (covering all grade levels K–5), district curriculum coordinators, and university educators across STEM+C content. This group met to collaborate in monthly face-to-face (in person and virtual) research meetings for three years (January 2019–August 2021) to determine areas of need for the continuous improvement of STEM+C in elementary schools.

Participants

Elementary school teachers in grades K–5 in a northeast state in the US participated in the Computational Thinking Survey through a web-based link that was sent through email to all public elementary schools in the state. Of the (N=560) teachers who participated in the survey, only teachers who teach science (N=259) from 30 different school districts' responses are used in this study. The demographics of these (N=259) teachers were as follows: 94.2% of them were females, 94.6% were white/Caucasian, 77.7% had over 10 years of experience, and only 30% of the teachers had more than three hours of computing professional development. The overall demographics of students being taught by these teachers in this state were 8% African American, 3% Asian, 25% Hispanic, 4% Multiracial, 1% Native American, and 59% White. From this population, 47.5% of the students were eligible for subsidized lunch and 15% of the population receive special education services (Count, 2018).

CT survey

The survey data used and analyzed for this study came from a portion of data related to science instruction, which was collected from a web-based cross-sectional CT survey administered to K–5 teachers in the late fall of 2019 and winter of 2020 using Survey Monkey software. The survey contained 55 questions and took teachers approximately 30 minutes to complete. The CT survey was examined for face and content validity by the DBIR team. A test-retest procedure was implemented to determine reliability using Spearman's rank-order correlation with a Spearman rho of 0.840, $p < .01$ (Laerd Statistics, 2018). The instrument was piloted with 125 elementary teachers from

March to June 2019, and minor changes were made to improve the questions, format, and frequency scales. Frequency levels of teaching time, including the number of days per week teaching science and the number of minutes usually spent on science lessons, were asked to get a general idea of how much time is devoted to science instruction in an elementary-level schedule. The survey also measured frequencies of use for each CT concept (algorithms, decomposition, abstraction, and pattern recognition) and approach (tinkering, creating, debugging, persevering, collaborating) by asking the percentage of time teachers spent on CT concepts and approaches while teaching science. Teachers were also asked to describe a lesson they taught in a sentence or two where they included the different CT concepts or approaches in the teaching and learning process. The survey was educative in describing the component parts of CT by using definitions, pictures, and examples of the different elements modified from Barefoot Computing at School curriculum (Barefoot Computing, n.d.). **Figure 1** shows an example of the description of the concept *patterns*, the questions asked, and how they appear in the survey.


Computational Thinking in Elementary School Classrooms

Concept: Patterns

Patterns are described as "spotting similarities and common differences. By identifying patterns we can make predictions, create rules, and solve more general problems" (Barefoot Computing, n.d.). When children learn to recognize repeating melodies in music or phrases in stories, when they see similarities and differences in data collected in science, or rules of number sequences in math, they are identifying patterns (www.barefootcas.org.uk).

OK

Students look for patterns to construct a Sierpinski Triangle.



(iStock.com/Nadezhda1906, 2019)

OK

* 27. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching patterns?

	Percent of lessons
ELA	<div></div>
Math	<div></div>
Science/Engineering	<div></div>
Social Studies	<div></div>
Specialization	<div></div>

28. In a sentence or two describe a lesson you taught that includes **patterns** in the teaching and learning process. (optional)

Figure 1

Example of survey description and questions for the CT concept patterns

Analysis

Descriptive data collected in the survey and the qualitative written responses to a question that asked teachers to describe a lesson they taught in a sentence or two where they included the different CT concepts (decomposition, abstraction, algorithms, or pattern recognition) or approaches (tinkering, debugging, creating, persevering, and collaboration) in the teaching and learning process were analyzed for this research paper

The qualitative responses were analyzed and coded by three members of the DBIR team for accuracy. Each member also identified the ten exemplar examples for each CT concept and approach. All the examples were classified by subject area, and only the responses sorted as science examples were used for this analysis. After examples were classified by subject, they were coded as being an accurate example of CT, an inaccurate example of CT, or unsure. The accuracy of the example was determined by the DBIR members, who had each spent two years studying CT, reading the examples, and comparing them to the shared definitions created through analysis of discourse, a standards crosswalk, and literature reviews. The definitions the DBIR team used are shown in Table 1.

After all the examples were analyzed and recorded by the three members, a fourth member identified each example as accurate or not based on at least 67% or more agreement. From this data, the subject and accuracy were quantified and analyzed.

Table 1

Example of CT Definitions

CT concept/approach	Teacher-friendly definition
algorithm	A sequence of steps to solve a problem or achieve an outcome
abstraction	Removing unnecessary details/information with a specific focus
pattern recognition	Spotting similarities and common differences; by identifying patterns students can make predictions, create rules, solve more general problems
decomposition	Refers to breaking something up into parts. In a computational thinking context, this means breaking a complex task into simpler subtasks.
tinkering	Using manipulatives to prove a theorem or property. Sticking with it until they do.
debugging	Finding mistakes and figuring out why something is not working.
creating	Using known/acquired information to design or construct and being able to share that creation with others.
perseverance	Sticking with it until the end, even if the deadline passes; intrinsic motivation
collaboration	Collaboration (as opposed to cooperation) involves mutual effort and contribution in pursuit of a shared goal. This effort and contribution do not have to be equal but is essential to achieving a creation of material or concept.

Results

The time spent on science instruction varied between districts in this state. Only 38% of the districts teach science every day, while 23% of districts teach science only one or two days a week. The typical length of time for science lessons also varies,

ranging from 20 minutes or less to more than 81 minutes. However, the majority of lessons are taught in the 20–40- or 40–60-minute range (Figure 2). Some districts are

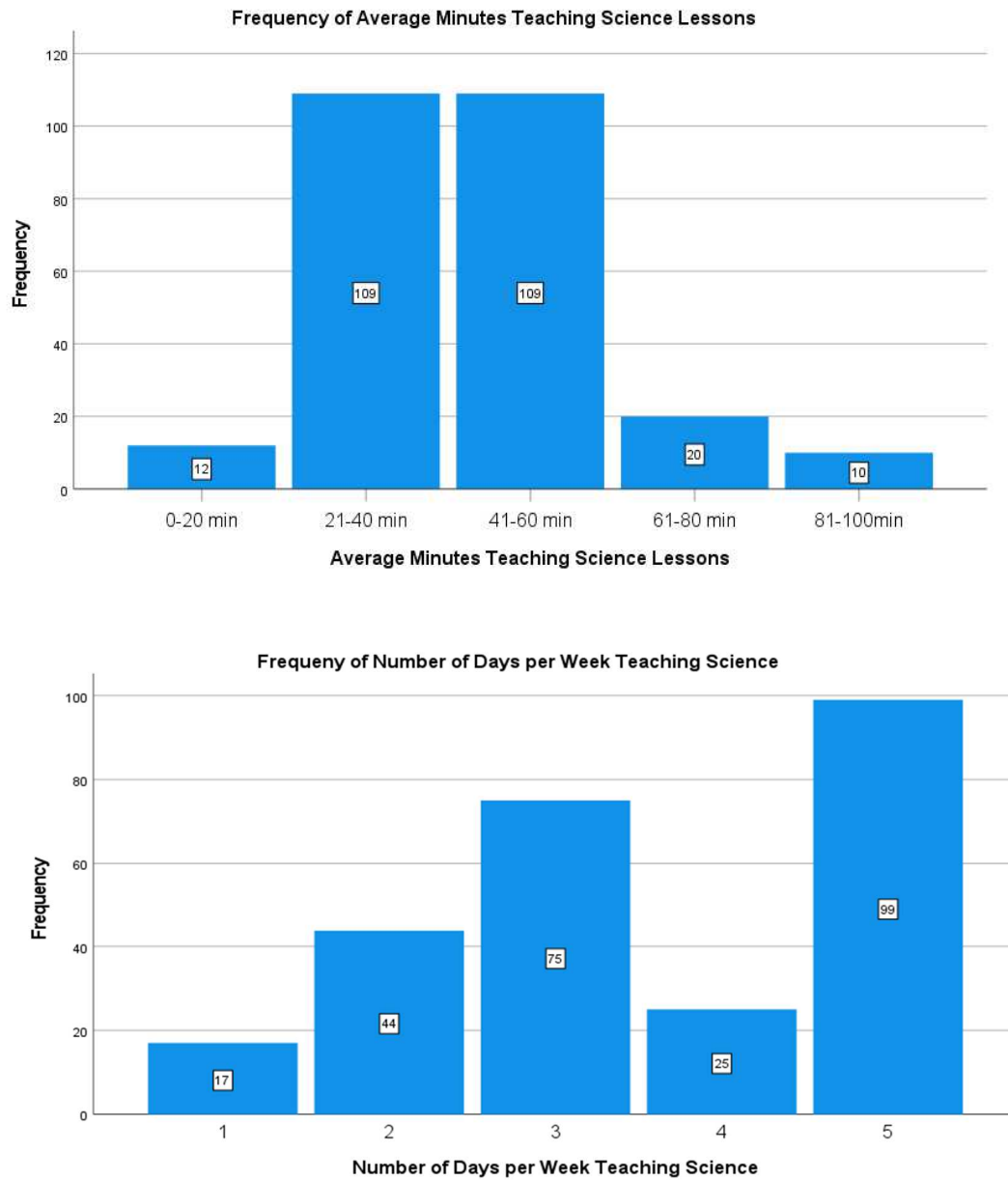


Figure 2

Time spent teaching science lessons

teaching every day with shorter durations, while some teach fewer days with longer duration. The state assumes science instruction is taught on a 100-day timeframe of three days per week with a timeframe of a 45–60-minute instruction block (Instruction & Assessment, n.d.). Most districts are meeting this requirement with the average weekly minutes for science being 151 minutes.

Computational Thinking concepts and approaches are currently being taught at different percentages of time in science instruction. Figure 3 shows the average percentage of time teachers report having the different computational thinking concepts and approaches present in their lessons. Based on this data, the concept with the largest average percentage of time spent on teaching and learning was decomposition with a mean percentage of 56.61%, followed by patterns (49.06%), algorithms (48.65%), and abstraction (44.94%).

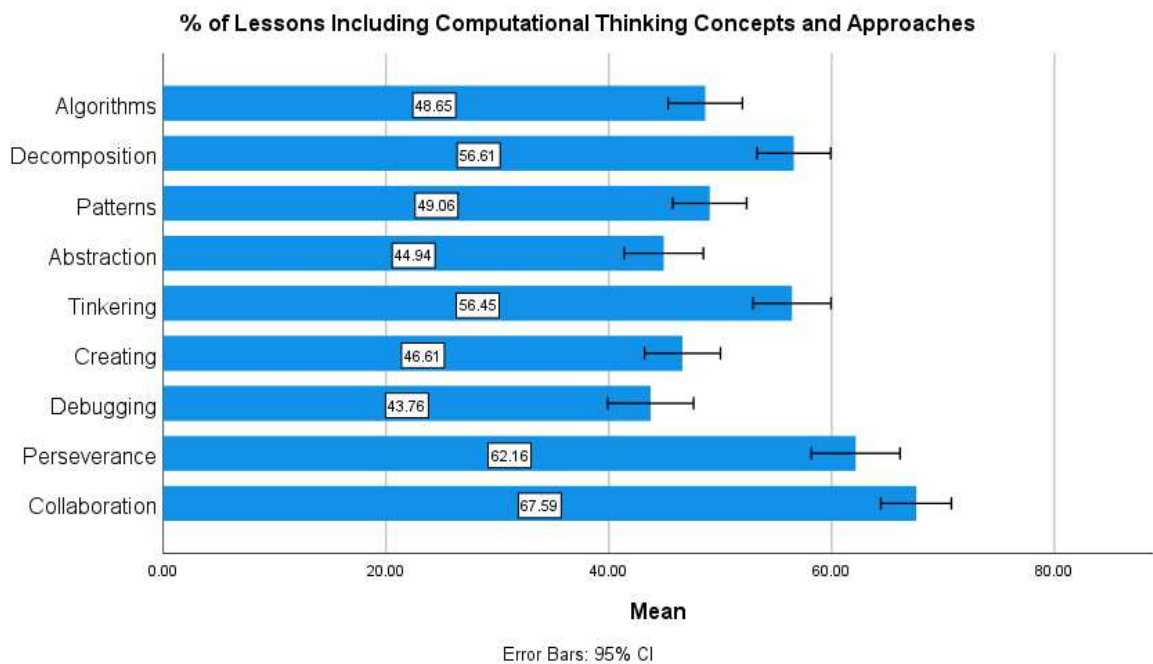


Figure 3

Percentage of Lessons Including CT Concepts and Approaches

Common ways of decomposing problems in computer science are to break problems down by structure, function, sequence, and dependence (Rich et al., 2019). Science instruction, which aligns with the state-adopted Next Generation Science Standards, often address structure and function which are one of the seven cross-cutting concepts. Cross-cutting concepts are core ideas that bridge disciplines in science (NGSS, 2013). Lessons are often designed to break down systems into component parts, whether structurally or functionally. Therefore, it is not surprising that decomposition was the concept used most. Patterns is also a cross-cutting concept in the NGSS standards, so it too is commonly used.

The concept with the least amount of time dedicated in science instruction was abstraction. This could possibly be explained by Jean Piaget's cognitive development theory which addresses how children are developing their schemas up until the age of 12 which makes it difficult for abstract thinking (Lister, 2011). In addition, Rijke et al. (2018) found older students do better with abstraction tasks than younger children. Since this survey is for the K–5 levels, teachers may focus more on building the foundational skills of thinking and less time on the higher-level thinking skills.

The approach with the highest average percentage of time spent was collaboration with 67.59%. This was followed by perseverance (62.16), tinkering (56.45), creating (46.61), and lastly debugging (43.76). An explanation for collaboration having the highest percentage of time is that science instruction often focuses on students engaging in social interaction, using scientific discourse, and applying scientific representations and tools while working in groups (National Research Council, 2012). It is not surprising

that teachers spent the least amount of time with debugging as it deals with higher-level cognitive thinking skills that may be developmentally less appropriate for this age group.

The results of the responses given for the examples of CT concept and CT approaches present in science lessons are in Table 2. Analysis of the CT concepts indicates that decomposition is the concept with the most examples at 27.2% of total examples. This aligns with teachers reporting decomposition was the concept they spend most of their time on. Patterns had a small number of examples (7.9%), but they were all accurate. Algorithms made up 13.8% of the examples. However, 36.8% of those examples were deemed inaccurate. Abstraction had the lowest percentage of examples for the CT concepts (6.2%). This aligns with the percentage of time teachers reported engaging in teaching and learning of these concepts in their instruction. Since they reported spending the most time teaching decomposition and least time with abstraction, it confirms their examples are proportional to their use.

Analysis of the CT approaches indicate that tinkering is the approach with 25.3% of total examples. This is followed by creating (7.6%), debugging (6.2%), collaboration (3.6%), and perseverance (2.2%). Teachers provided 70 examples of tinkering, which were the most examples out of the five approaches. Interestingly, teachers had the greatest accuracy (99%) with tinkering. An example of a teacher description for tinkering is *“In science the students were given a light bulb, wires, and a battery. They have to make a light bulb light with their materials.”* This provides evidence that the definition, example, and picture for tinkering were noticeably clear to the teachers because they were able to easily identify how a student uses manipulatives to prove a property by

sticking with it. Teachers were able to identify tinkering happening in their practice 56.5% of their time.

Among all CT concepts, teachers spent 48.6 % of their time on algorithms and provided many examples (14% of total examples) that were the least accurate. This may be an example of a concept that teachers are not as clear at identifying in their practice. For example, the following is an inaccurate example of a teacher describing in their own words the CT concept of algorithms: *“During STEM activities students are given a problem to solve with their group. Each group is provided with the same materials; they may use all or some of the materials but may not use any other materials or extra materials.”* This is an overly broad example that does not explicitly talk about the organization and identification of steps the students underwent in this STEM activity, so it was deemed inaccurate.

Collaboration and perseverance are the CT approaches teachers spent most time engaging in teaching and learning (67.7%, 62.2%) but provided the least number of examples (4%, 2%). Interestingly, teachers provided quite accurate examples for collaboration but less accurate examples for perseverance. An inaccurate example for perseverance is *“Science investigations and engineering challenges require students to persevere through a task”* This teacher example lacks details explaining what exactly the child is doing in this lesson. The small number of examples in science can be explained by the fact that most examples reported by elementary teachers for collaboration and perseverance from the survey were identified as happening in ELA and math. Since elementary teachers spend the majority of their time teaching these subjects and these approaches integrate well in all subject areas, this explains the lower number of examples

collected.

The majority of examples given for CT were the concepts with 152 examples (55.1%) versus 124 examples (44.9%) for the approaches; however, the examples given for CT approaches were more accurate (93.5%) than the concepts (86.0%).

Table 2-

Examples of CT Concepts and Approaches

CT Concepts	Number of Science Examples	Accuracy	% of Total Examples
Decomposition	75	97.3%	27.2%
Algorithms	38	63.2%	13.8%
Abstraction	17	70.6%	6.2%
Patterns	22	100%	7.9%
CT Approaches	Number of Science Examples	Accuracy	% of Total Examples
Tinkering	70	98.6%	25.3%
Creating	21	95.2%	7.6%
Debugging	17	82.4%	6.2%
Perseverance	6	66.7%	2.2%
Collaboration	10	90.0%	3.6%
Computational Thinking	Total Number of Science Examples	Average % Accuracy	% of Total Examples
Concepts	152	86.0%	55.1%
Approaches	124	93.5%	44.9%

In addition to determining accuracy of examples of CT in the teaching and learning of science, there are qualitative examples the research team found to be good representative examples. In Table 3, examples of CT concepts and approaches taken from the survey are displayed showing examples that were in 100% agreement for accuracy and rated as top examples by the DBIR team based on the simplified definitions created by the DBIR team. In addition are examples that were deemed inaccurate by the DBIR team.

Table 3

Accurate and Inaccurate Examples of CT Concepts and Approaches

Computational Thinking Teacher Accurate Examples and Inaccurate Examples of Concepts		
CT Concepts	<i>Accurate Examples</i>	<i>Inaccurate Examples</i>
Decomposition	<i>“Creating a food web in science”</i>	<i>“Deciding on a research topic”</i>
Patterns	<i>“Looking across weather (temperature) data plotted on a color-coded graph, students identify patterns of cold/warm weather periods.”</i>	<i>“When teaching changing states of matter, we analyzed patterns to help students decipher the difference between physical and chemical changes.”</i>
Algorithms	<i>“In science, students were taught to follow specific directions in a specific order to sort rocks, gravel, and sand by size using screens.”</i>	<i>“During STEM activities students are given a problem to solve with their group. Each group is provided with the same materials; they may use all or some of the materials but may not use any other materials or extra materials.”</i>
Abstraction	<i>We use graphic organizers to record the essential information about an animal group (ex. amphibians).</i>	<i>“Students plan how they will observe shadows and then collect data to determine if the hypothesis is true.”</i>

Computational Thinking Teacher Accurate Examples and Inaccurate Examples of Approaches		
CT Approaches	Accurate Examples	Inaccurate Examples
Collaboration	<i>“Science investigation of the effect of rain on natural materials -- students worked in groups of four to predict, test, observe and write about their observations. One child was in charge of materials, one was “taskmaster” to make sure everyone was participating, one was “crowd control” to make sure they focused and kept their voice level at the correct volume, and one was in charge of cleaning up.”</i>	<i>“All science projects”</i>
Creating	<i>“Building a cardboard house and using skills from electricity to light the rooms.”</i>	<i>“For our bug unit, students create a culminating project which shows what they have learned about 5 research questions”.</i>
Debugging	<i>“We are teaching Physical Science--Pushes and Pulls. The children are asked to Launch their ball to reach a chosen target. Children soon realize that they have to modify/debug the force given and starting position to hit the target. This continues until they complete their task.”</i>	<i>“Phases of the Moon”</i>
Perseverance	<i>“When creating a structure to hold a book, we needed multiple tries and perseverance”</i>	<i>“Science investigations and engineering challenges require students to persevere through a task”</i>
Tinkering	<i>“In science the students were given a light bulb, wires, and a battery. They have to make a light bulb light with their materials.”</i>	<i>“Science - all lessons are hands on”</i>

Summary of Findings and Policy Implications

Integrating CT into science instruction is one pathway to improve teaching and learning in a technologically rich society without adding one more thing to an elementary teacher's plate. Integrating CT into a subject all students take allows more diverse students access to skills needed in the digital age so they can be informed users, creative thinkers, and producers of technological innovations (Weintrop, 2016). When implementing CT into K–5 education systems, it is critical to make sure all stakeholders involved understand the purpose of why this change should be made. Policies need to be put into place to provide a framework in order for computer science goals to be achieved. Districts need to select high quality and evidence-based programs to support student learning, and teaching practices need to be put in place to make sure outcomes are effective in providing students access to foundational knowledge in computer science.

This study sought to add to the literature on teacher practices for implementing CT into science instruction by analyzing data and written responses collected about teachers' current practices. The research questions included:

- 1) How often and how long do K–5 teachers teach science?
- 2) What percentage of time do teachers report teaching CT concepts and approaches in their science classes?
- 3) How accurate are CT concepts and approaches in the examples given by K–5 teachers currently taught in science classrooms?
- 4) How do teachers describe lessons where they include the different CT concepts in the teaching and learning process?

The answers for the first research question, concerning how often science teachers teach science during the week and the length of the typical lesson, varied between teachers and districts. Most schools are not teaching science every day, and most lessons have a length of an hour or less, resulting in an average of 151 minutes per week of teaching time. Most time in an elementary school day is devoted to math (321 average minutes/week) and ELA instruction (341 average minutes/week). With CT becoming an additional skill we want all of our children to have, we need to consider increasing the length of time science is being taught. Curran & Kitchin (2019) analyzed data from the Early Longitudinal Childhood Study and found that more time spent teaching science increased academic achievement in science because it provided more opportunities to learn. Prioritizing what is most important for students to learn is something districts need to reflect on constantly with our rapidly changing world.

In answering the second research question (What is the percentage of time teachers report teaching CT concepts and approaches in their science classes?), analysis revealed that teachers find opportunities for all the concepts and approaches in their teaching. Analyzing the data showed science teachers spent most of their time teaching students how to decompose or break down problems and the least amount of time on abstraction. Abstraction is a higher-level thinking concept, so it might be a concept teacher do not teach in the lower elementary level as frequently because their students are still developing their base knowledge. Patterns had the second highest frequency of use for the concepts which could be a result of the overlap of the cross-cutting concepts of patterns listed in the NGSS in addition to CT, which is listed in the engineering practices.

Teachers also identified different CT approaches during science lessons, with much of their instructional time engaging students in collaboration. A great deal of inclusive science instruction focuses on students engaging in social interaction, using scientific discourse, and applying scientific representations and tools while working in groups, so this is not an approach that is new to science instruction (National Research Council, 2012). Debugging is used the least which may be a result of this skill not being as developmentally appropriate for this age group as the other approaches. On the other hand, perhaps debugging along with creating are approaches teachers need encouragement for learning more about through professional development.

In answering the third research question, concerning how accurate CT examples are given by K–5 teachers, the analysis revealed when teachers are provided definitions, pictures, and examples, they are able to accurately describe potential areas of where the concepts and approaches are present in their lessons. Teachers were accurate with CT concepts 86% of the time and 93.5% of the time with approaches. This data shows that there are many opportunities for teachers to integrate CT in their lessons at the “Exist” stage of Waterman’s framework where they can explicitly teach the thinking processes that are happening in the lesson. However, future research needs to determine what CT practices should be highlighted at the Exist level in science class, and professional development should provide the expertise on how to integrate those key concepts or approaches.

Lastly, when answering research question number four, concerning how teachers describe a lesson where they included the different CT concepts (decomposition, abstraction, algorithms, or pattern recognition) or approaches (tinkering, debugging,

creating, persevering, and collaboration) in the teaching and learning process, a myriad of examples were found, with highlights shown in Table 3. Examples of lessons were present in the domains of Life Science, Physical Science, Earth and Space Science, and Engineering, Technology and Applications of Science. However, the examples given were shared independently and not connected to other concepts and approaches as they typically would be taught in lessons. In order to have teachers understand how to implement these into their instruction, it would be useful for professional development providers to supply a better picture of the daily classroom showing how these concepts and approaches are interrelated. One area of improvement this DBIR team concentrated on after analyzing the survey was developing concrete examples of CT-integrated instruction teachers could use to help with implementation efforts, using vignettes of what CT would like in a typical science lesson (Appendix A). These vignettes serve as additional examples of CT concepts and approaches that exist in science curriculum. Yadav et al. (2018) contends vignettes are a useful tool to use as a way of measuring change in teachers thinking about CT.

The first steps in getting teachers interested in integrating CT into their instruction are to make them aware of what computational thinking is and having district leaders make it a school-wide goal (Mills et al., 2021). This CT survey served as a tool for spreading awareness because at the end of the survey when teachers were asked if CT should be a priority in schools, 76.5% agreed or strongly agreed, and 80.8% believe it is relevant to their work. Opportunities for teaching CT concepts and approaches are present in curriculum all throughout this northeast state. The next steps would be to explicitly teach the CT concepts and approaches using methods such as framing,

prompting, and inviting reflection, which would allow science students and teachers to start sharing a common language with computer science (Rich et al., 2021). Once the school is confident with the “Exist” stage they can begin working towards the “Enhance” and “Extend” in future lessons.

This analysis also gave information about how CT concepts and approaches are present in a sizable portion of the teaching time spent on science instruction. The average teaching time for science was determined to be 151 minutes per week, which is similar to the national average, but only half the amount of time recommended by the National Science Teachers Association (NSTA). The NSTA recommends at least 60 minutes per day (Sparks, 2021). Given the importance of CT skills and knowing how easily it integrates into science we need to consider increasing the length of time science is being taught.

Limitations and suggestions

Like all research, there are limitations that should be considered as the results from this study are interpreted. This study used the definitions of CT developed by Barefoot Computing and the following work of the DBIR team. Although these definitions are developed with the goal of teacher-friendliness, it is recognized that there still is no agreed upon definition in the literature. Moreover, the definitions may change as the needs and contexts change. Therefore, it is important that the computer science and education communities come together and decide on a shared terminology of what computational thinking is along with the skills involved. By having shared language, measuring progress on implementation efforts can be more accurately assessed.

The second limitation is that when teachers were asked to describe their lessons that included the different CT elements, teachers described the concepts and approaches independently, whereas a typical lesson might have multiple concepts and approaches happening within the same lesson. Therefore, some of those interrelationships are missed. Future professional development work might consider using vignettes that have multiple CT concepts and approaches (as in Appendix A) embedded in the science lesson to serve as examples for measuring change in teachers thinking about CT.

Although a detailed description of each concept and approach with definitions, pictures, and examples was given to provide a shared understanding, the percentage of time engaging in CT skills is self-reported by teachers. Therefore, in the future, it is recommended to collect qualitative observational data to examine how teachers integrate CT skills in their instruction to have a more holistic view.

Lastly, although representation from all districts was included in the data collected, a limitation in sampling most likely occurred because teachers who are more motivated and interested in CT and research might have participated in the survey. Future research is still needed on how effective CT-integrated instruction affects the learning of CT and science and how it affects students' interests and attitudes in computer science.

References

- Adler, R. F., & Kim, H. (2018). Enhancing future K-8 teachers' computational thinking skills through modeling and simulations. *Education and Information Technologies*, 23(4), 1501-1514. <https://doi.org/10.1007/s10639-017-9675-1>
- Angeli, C., & Jaipal-Jamani, K. (2018). Preparing pre-service teachers to promote computational thinking in school classrooms. In *Computational thinking in the stem disciplines: Foundations and research highlights* (pp. 127-150). Springer, International Publishing.
- Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Journal of Educational Technology & Society*, 19(3), 47-57.
- Barefoot Computing. (n.d.). Retrieved from <https://www.barefootcomputing.org>
- 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/10.1145/2676723.2693616>
- Berry, M. (2013). *Computing in the National Curriculum: A Guide for Primary Teachers*, Computing at School, Bedford.
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education-Implications for policy and practice* (No. JRC104188). Joint Research Centre (Seville site).
- Boulden, D. C., Wiebe, E., Akram, B., Aksit, O., Buffum, P. S., Mott, B., Boyer, K. E., & Lester J. (2018). Computational thinking integration into middle grades science classrooms: Strategies for meeting the challenges. *Middle Grades Review*, 4(3) 1-17. <https://scholarworks.uvm.edu/mgreview/vol4/iss3/5>
- Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017, November). Development of computational thinking skills through unplugged activities in primary school. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 65-72). <https://doi-org.uri.idm.oclc.org/10.1145/3137065.3137069>
- Bureau of Labor Statistics (2021). U.S. Department of Labor, Computer, and Information Research Scientists: *Occupational Outlook Handbook*. Retrieved from <https://www.bls.gov/ooh/computer-and-information-technology/computer-and-information-research-scientists.htm>
- Bybee, R. W. (2013). *The case for STEM education: Challenges and opportunities*. NSTA Press.
- CAST (2018). Universal Design for Learning Guidelines version 2.2. Retrieved from

<http://udlguidelines.cast.org>

- Chalmers, C. (2018). Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17, 93-100. <https://doi.org/10.1016/j.ijcci.2018.06.005>
- Coburn, C. E., & Penuel, W. R. (2016). Research–practice partnerships in education: Outcomes, dynamics, and open questions. *Educational Researcher*, 45(1), 48-54. <https://doi.org/10.3102/0013189X16631750>
- Coburn, C. E., Penuel, W. R., & Farrell, C. C. (2021). Fostering educational improvement with research-practice partnerships. *Phi Delta Kappa*, 102(7), 14-19. <https://doi.org/10.1177/00317217211007332>
- Code.org, CSTA, & ECEP Alliance. (2021). 2021 State of computer science education: Accelerating action through advocacy. Retrieved from <https://advocacy.code.org/stateofcs>
- Count, R. I. (2018). Policy & Advocacy for Rhode Island’s Children. Retrieved from <http://www.rikidscount.org/>
- Curran, F. C., & Kitchin, J. (2019). Early elementary science instruction: Does more time on science or science topics/skills predict science achievement in the early grades? *AERA Open*, 5(3), 2332858419861081.
- CSforAll (2022). Computer Science for All. January 1, 2022, from <https://www.csforall.org>
- CS4RI (2022). Retrieved January 1, 2022, from <https://www.cs4ri.org/ri-cs-education-standards>
- Dasgupta, A., Rynearson, A. M., Purzer, S., Ehsan, H. and Cardella, M. E. (2017). Computational thinking in K-2 Classrooms: Evidence from student artifacts (Fundamental). A paper presented at American Society for Engineering Education Annual Conference and Exhibition, June 25-28, 2017, Columbus, Ohio. <https://doi.org/10.18260/1-2--28062>
- Denning, P. J., & Tedre, M. (2019). *Computational Thinking*. MIT Press.
- Every Student Succeeds Act, 20 U.S.C. § 6301 (2015). <https://www.congress.gov/114/plaws/publ95/PLAW-114publ95.pdf>
- Fisher, L. M. (2016). A decade of ACM efforts contribute to computer science for all. *Communications of the ACM*, 59(4), 25–27. <https://doi.org/10.1145/2892740>
- Gane, B. D., Israel, M., Elagha, N., Yan, W., Luo, F., & Pellegrino, J. W. (2021). Design and validation of learning trajectory-based assessments for computational thinking

- in upper elementary grades. *Computer Science Education*, 31(2), 141–168.
- Grover, S. (2018). The 5th ‘C’ of 21st century skills. *Try computational thinking (not coding)*. (March 13). Retrieved from EdSurge News: <https://www.edsurge.com/news/2018-02-25-the-5th-c-of-21st-century-skills-try-computational-thinking-not-coding>.
- Hsu, Irie, N. R., & Ching, Y.-H. (2019). Computational Thinking Educational Policy Initiatives (CTEPI) Across the Globe. *TechTrends*, 63(3), 260–270. <https://doi.org/10.1007/s11528-019-00384-4>
- Instruction & Assessment. RI Model Science Curriculum. (n.d.). Retrieved January 19, 2022, from <https://www.ride.ri.gov/InstructionAssessment/Science/>
- Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, 82, 263-279. <https://doi.org/10.1016/j.compedu.2014.11.022>
- ISTE (2011). Computational Thinking in K–12 Education leadership toolkit. *Computer Science Teacher Association*: http://csta.acm.org/Curriculum/sub/CurrFiles/471.11_CTLeadershipToolkit-SP-vF.pdf *adresinden alindi*.
- ISTE Standards for Educators (2022). Retrieved January 1, 2022, from <https://www.iste.org/standards>
- Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers’ self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, 26(2), 175–192. <https://doi-org.uri.idm.oclc.org/10.1007/s10956-016-9663-z>
- K–12 Computer Science Framework. (2016). Retrieved from <http://www.k12cs.org>
- Karpinski, Z., Biagi, F., & Di Pietro, G. (2021). Computational Thinking, Socioeconomic Gaps, and Policy Implications. IEA Compass: Briefs in Education. Number 12. *International Association for the Evaluation of Educational Achievement*.
- Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2019). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, 1-15. <https://doi.org/10.1007/s10956-019-09798-4>
- Laerd Statistics (2018). Spearman's correlation using SPSS Statistics. *Statistical tutorials and software guides*. Retrieved from <https://statistics.laerd.com/>
- Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2019). Computational thinking from a disciplinary perspective: Integrating computational thinking in K–

- 12 Science, Technology, Engineering, and Mathematics education. *Journal of Science Education and Technology*, 29(1), 1-8. (2020) 29:1–8. <https://doi.org/10.1007/s10956-019-09803-w>
- Lee, I., & Malyn-Smith, J. (2020). Computational Thinking Integration Patterns Along the Framework Defining Computational Thinking from a Disciplinary Perspective. *Journal of Science Education and Technology*, 29(1), 9–18. <https://doi.org/10.1007/s10956-019-09802-x>
- Lister, R. (2011). Concrete and other neo-piagetian forms of reasoning in the novice programmer. *Conferences in Research and Practice in Information Technology Series*, 114, 9–18.
- Maltese, A. V., & Tai, R. H. (2010). Eyeballs in the fridge: Sources of early interest in science. *International Journal of Science Education*, 32(5), 669-685. <https://doi-org.uri.idm.oclc.org/10.1080/09500690902792385>
- Martínez-García, E. (2021). Computational thinking: the road to success in education. *Academia Letters*, Article 3973. <https://doi.org/10.20935/AL3973>
- Mills, K., Coenraad, M., Ruiz, P., Burke, Q., & Weisgrau J. (2021, December). Computational thinking for an inclusive world: A resource for educators to learn and lead. *Digital Promise*. <https://doi.org/10.51388/20.500.12265/138>
- Montoya. (2017). Computer Science for All: Opportunities Through a Diverse Teaching Workforce. *Harvard Journal of Hispanic Policy*, 29.
- National Research Council. (2012). *A framework for K–12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press.
- NGSS Lead States. (2013). Next generation science standards: For states, by states. Washington, D.C.: National Academies Press.
- NSTA. (n.d.). *About the next generation science standards*. NGSS@NSTA. Retrieved December 26, 2021, from <https://ngss.nsta.org/about.aspx>
- Ouyang, Y., Hayden, K. L., & Remold, J. (2018, February). Introducing Computational Thinking through Non-Programming Science Activities. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 308-313). ACM. <https://doi.org/10.1145/3159450.3159520>
- Prottsman, K. (2019). *Computational Thinking Meets Student Learning: Extending the ISTE standards*. International Society for Technology in Education.
- Rhode Island Computer Science Education Standards. (2018). <https://www.cs4ri.org/ri-cs-education-standards>

- Rich, P. J., Jones, B., Belikov, O., Yoshikawa, E., & Perkins, M. (2017). Computing and engineering in elementary school: The effect of year-long training on elementary teacher self-efficacy and beliefs about teaching computing and engineering. *International Journal of Computer Science Education in Schools*, 1(1), 1-20. <https://doi.org/10.21585/ijcses.v1i1.6>
- Rich, Egan, G., & Ellsworth, J. (2019). A Framework for Decomposition in Computational Thinking. Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education, 416–421. <https://doi.org/10.1145/3304221.3319793>
- Rich, K. M., Yadav, A., & Larimore, R. A. (2020). Teacher implementation profiles for integrating computational thinking into elementary mathematics and science instruction. *Education and Information Technologies*, 25(4), 3161–3188. <https://doi.org/10.1007/s10639-020-10115-5>
- Rhode Island Board of Education Act, RIGL § 16-22- 31(2021). <https://law.justia.com/codes/rhode-island/2021/title-16/chapter-16-22/section-16-22-31/>
- Ruberg, L. F., & Owens, A. (2017). A future-focused education: Designed to create the innovators of tomorrow. In *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 367-392). Springer, Cham.
- Ryoo, J. J. (2019). Pedagogy that supports computer science for all. *ACM Transactions on Computing Education (TOCE)*, 19(4), 36. <https://doi.org/10.1145/3322210>
- Sengupta, P., Kinnebrew, J., Basu, S., Biswas, S., & Clark, G. (2013). Integrating computational thinking with K–12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18(2), 351-380. <https://doi.org/10.1007/s10639-012-9240-x>
- 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>
- Sweetman, S. (Principal Investigator). (2018-2020). *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (Award number: 1813224) [Grant] National Science Foundation.
- 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(1), 53-64. <https://doi.org/10.1007/s10956-019-09801-y>
- Webb, M., Davis, N., Bell, T., Katz, Y. J., Reynolds, N., Chambers, D. P., & Syslo, M. M. (2017). Computer science in K–12 school curricula of the 21st century: Why,

what, and when? *Education and Information Technologies*, 22(2), 445-468.
<https://doi.org/10.1007/s10639-016-9493-x>

- Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1), 127-147.
<https://doi.org/10.1007/s10956-015-9581-5>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
<https://doi.org/10.1145/1118178.1118215>
- Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In *Emerging research, practice, and policy on computational thinking* (pp. 205-220). Springer, Cham.
- Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K–12 classrooms. *TechTrends*, 60(6), 565–568. <https://doi.org/10.1007/s11528-016-0087-7>
- Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, 14(1), 5, 1-16. <https://doi.org/10.1145/2576872>

Appendix A

Concept or Approach	Grade level	Vignette
Decomposition Abstraction Patterns Debugging Creating Collaboration	First Grade	<p style="text-align: center;">CT Vignette</p> <p>First grade students at Captain Isaac Paine Elementary completed a design challenge asking them to design a parachute system that could successfully deliver supplies (goldfish crackers) to the ground. Throughout the lesson, students engaged in the engineering design process and utilized a variety of Computational Thinking (CT) concepts and approaches. To learn more about how parachutes worked, students conducted research. They watched a short informational video on parachutes and as a class collected evidence explaining how a parachute works. The process of selecting only the evidence that answered their research question demonstrates students engaging in the CT concept of abstraction. This CT concept was also present during the planning phase of the design challenge. Students made decisions about which materials were the most important and included them in their plans while leaving out those that were deemed unnecessary. When students were ready to create their designs, groups intuitively relied on the CT concept of decomposition to build the parachute system in parts. Many groups started with building a parachute canopy that would fill with as much air as possible. Then they worked on constructing a stable base that would hold the supplies (goldfish crackers) when it landed. Finally, the students had to figure out a way to securely attach the canopy to the base. Breaking down the design into smaller parts made a complex task more manageable for the students. When it was time to test their designs, each group's parachute was dropped from a high location. Classmates collaborated with each other by providing feedback on how parachute designs could be improved. Analyzing the patterns they observed from testing different parachute designs allowed groups to debug and try to fix any problems with their design.</p>
Abstraction Pattern recognition Collaboration Creativity algorithms	Second grade	<p>Second grade students at Narragansett Elementary School follow the Foss curriculum for Insects and Plants. The following is a unit created to find what structures all insects have as opposed to structures that some insects may have. Students worked in pairs to brainstorm a list of what they believed were insects on a Padlet. They pulled from the knowledge they have gained through the start of our insect unit in science. Students then picked 5 insects from a list the class created. Students then researched, using kid search engines, these insects to see if they are true insects. Students then created a list of the commonalities their insects have. Next, students narrowed down the list further to the structures that all insects have. Example: head, thorax, and abdomen all insects have, but only some have wings. Students then designed a business card that outlines a set of steps to help a family member determine what is and is not an insect.</p>
Decomposition	Third	Grade 3 science- Isabella, a third-grade student in an elementary

<p>Recognizing patterns Abstraction Algorithms Creating collaborating</p>	<p>Grade</p>	<p>classroom in Warwick, Rhode Island, is learning about the life cycles of plants and animals in a science classroom. She has been tasked with investigating how different animals and plants undergo a series of changes in their lifetime. She has the choice of using a computer, various books, interactive games, or information cards. She has decided to research a frog and an oak tree. She learns about the frog by reading the story <i>Tale of a Tadpole</i> by Karen Wallace and learns about an oak tree's life cycle by playing a memory game with her friend Luca. She uses decomposition when she finds out the frog life cycle can be broken down into distinct stages, which include eggs, embryo, tadpole, tadpoles with 2 legs, tadpoles with 4 legs, and eventually a frog. She also uses decomposition with the oak tree when she finds out about the acorn, sprout, sapling, and tree.</p> <p>After students have had time to conduct their research, Mrs. Smith brings all the students together to present what they have found through their research. She asks them what all the organisms have in common. Robbie raises his hand and says, "They all start out small by being born, grow bigger, make more, and then die." Robbie has just demonstrated recognizing patterns. After a discussion, they all learn that organisms go through birth, growth, reproduction, and death. The students are then tasked with looking at the different stages they have identified and brainstorm together a list of characteristics that make each stage unique. Violet shares with her group that she notices a lot of the animals presented by her classmates have a stage where there is an egg that turns into some kind of larva. Aubrey says she notices the same thing with plants, but instead of some type of egg there is a seed that ends up sprouting. These girls have just used abstraction where they have pulled out the main information and removed the unnecessary details of these complex life cycles. The teacher then passes out 2 paper plates to each student and asks the students to choose a plant or animal they researched and draw the distinct stages in order in a circle on one plate. This is an example of students creating algorithms or a sequence of steps to achieve an outcome. The other plate goes on top of the life cycle plate with a window cut out that spins so students can only view one stage at a time. Students share their creations with their classmates and make predictions about what stage they think will be next when viewing each other's work. Students enjoy creating and collaborating together during the process of learning different life cycles.</p> <p>3-LS1- 1. Develop models to describe those organisms have unique life cycles but all have in common birth, growth, reproduction death.</p> <p>(Create student work here)</p> <div data-bbox="678 1692 1430 1850"> </div>
---	--------------	---

		Lesson using NICERC (National Integrated Cyber Education Research Center) Computational Thinking 3rd Grade Science Curriculum
Abstraction Tinkering Debugging	Fifth Grade	<p>Grade 5 Science</p> <p>Mixtures and Solutions kit: Students are tasked to determine the inside design of a black box, which is permanently taped closed. They need to decide what shape the object is inside and where it is located. Once they agree on a design, the students then create a model of what they believe to be a replica of the box they have and evaluate their work. At this point, they may revise their work and continue to try until they come to a consensus.</p> <p>This lesson is primarily designed to teach the students those models are explanations of objects, events, or systems that cannot be observed directly. That they are used to communicate and test and that, while developing their own, they could observe, construct, analyze and revise, multiple times over.</p>

CHAPTER 5

CONCLUSION OF COMPUTATIONAL THINKING IN ELEMENTARY SCIENCE

Elementary teachers have many demands on their time during the school day and figuring out efficient ways to incorporate computer science skills students need to live, play, and work in the 21st century are essential. Integrating computational thinking (CT) into science instruction is one way to make sure all students get the foundational skills they need without adding one more thing to a teacher's plate.

The purpose of the research conducted for this dissertation was to explore CT integration in K–5 science classrooms in a northeast U.S. state. The goal is to bring knowledge and awareness of computational thinking in elementary science to elementary teachers, administrators, curriculum developers, researchers, and policy makers by learning about descriptions of CT and the different concepts and approaches involved with it. In addition, this research aims to gain a better understanding of the computational thinking concepts and approaches that are occurring in science classrooms across a northeast state and the different variables that predict integration within instruction. This information will be used to determine areas of need for implementation.

The following research questions guiding the research for this dissertation are:

- 1) What is the definition of computational thinking and the skills involved?
- 2) What factors (teachers and district characteristics) are related to the use of CT skills?
- 3) How are elementary teachers using CT skills in their science classrooms?

Summary of Findings

In answering the first research question concerning what the definition of computational thinking and the skills involved are, which is addressed in the first paper, a literature review on how computational thinking is defined in K–12 schools was first conducted. After examining numerous definitions (Barr and Stephenson, 2011; ISTE and CSTA, 2011; NGSS, 2013; Wing, 2006), it was decided to refine the literature review to only articles that used computational thinking in the context of being integrated into elementary science. From this review only 20 articles met the requirement of the literature search. This literature review resulted in there being no agreed upon definition for elementary science, but eleven different definitions with different component parts. The different component parts also had varying definitions which makes integrating CT difficult when deciding how to best implement CT into elementary instruction. However, there was consensus across all articles that CT is a skill that needs to be implemented in elementary instruction.

Based on the findings in the literature review, research was conducted on creating a shared definition for CT and its component parts for the elementary level by a Design-Based Implementation Research (DBIR) Team, which included myself. From these three years of working together the team created definitions for CT and its component parts that will help with supporting elementary teachers in understanding CT. These definitions will also provide understandable language to help curriculum designers, policy makers, and researchers create programs that are specifically aimed and appropriate for elementary classrooms.

In answering the second research question, which is found in the second manuscript, concerning what the factors (teachers and district characteristics) are related to the use of CT skills, data from a CT cross-sectional web-based survey was analyzed that measured frequencies of use for each CT concept (algorithms, decomposition, abstraction, and pattern recognition) and approach (tinkering, creating, debugging, persevering, collaborating) by asking the percentage of time teachers spent on CT concepts and approaches while teaching science. The survey also measured teacher-level and district-level factors. The teacher factors measured were years of experience teaching, grade level taught, the amount of time spent engaging in computer science professional development, levels of confidence teaching CT, and teachers' levels of concern for implementing CT in the classroom. The district variables of interest include the district classification of the urban, urban ring, rural, or suburban, socioeconomic status based on the percentage of families below the poverty level, and whether the district belongs to an RPP.

This quantitative study is unique because based on the literature review on the integration of CT in science, most of the studies are exploratory and mostly qualitative with small samples. This study used a large sample from over 250 teachers in thirty districts. This study is a quantitative study which used Hierarchical Linear Modeling (HLM), to allow for generalizing and replicating findings across multiple levels (Creswell, 2014). HLM also is more efficient at accounting for variance among variables at different levels than other existing analyses and thus allows us to consider impacts in the complex systems of education (Woltman et al., 2012).

From this analysis, grade level taught, and teacher confidence levels were significant predictors for all the CT concepts and approaches. Teachers with more years of experience are more likely to engage in science activities that include the CT approaches of debugging and collaboration and the CT concepts of recognizing patterns and performing abstraction. The amount of computer science professional development did not predict the CT usage or the level of concern of the teacher. However, most teachers had little professional development with only 30% of teachers having above three hours. The same patterns occurred with levels of concern. The majority of the teachers (61%) were at the beginning stages of concern.

Belonging to a RPP was a significant district predictor for using algorithms, debugging and perseverance. Classification of districts was not a predictor of CT usage. Socioeconomic status based on the percentage of families below the poverty level was a predictor when analyzed independently of the RPP for algorithms, abstraction, creating, debugging, perseverance, and collaboration. The higher the percentage below the poverty level the less usage of the CT concepts and approaches. Several variables that predict CT usage from this study can help inform policy makers, professional development providers and curriculum designers so intentional implementation of CT integrated science lessons can support early computer science thinking skills.

In answering the third research question concerning how elementary teachers are using CT skills in their science classrooms, which is answered in the third paper, data from the same CT cross-sectional web-based survey was analyzed. Frequency levels of teaching time including the number of days per week teaching science and

the number of minutes usually spent on science lessons was asked to get a general idea of how much time is devoted to science instruction in an elementary level schedule. In addition, teachers were also asked to describe a lesson they taught in a sentence or two after they read definitions, viewed pictures, and were given basic examples of the different elements modified from Barefoot Computing at School curriculum (Barefoot Computing, n.d.).

The results revealed that the time spent teaching science on average is half the amount of time spent on ELA and math instruction. However, the results indicated that the percentage of lessons that have CT concepts (algorithms, decomposition, abstraction, pattern recognition) and approaches (tinkering, creating, debugging, persevering, collaborating) present in the science instruction are happening quite frequently. Teachers also gave numerous examples of lessons where they saw opportunities for integrating CT concepts and approaches within their lessons and were accurate with their descriptions. The information from this study is helpful to elementary teachers, administrators, curriculum developers, researchers, and policy makers because it reveals that there are many opportunities for teachers to integrate CT in their lessons at the “exist” stage of Waterman’s (2020) framework which is a helpful framework that uses Exist, Enhance, and Extend when integrating computational thinking.

After analyzing the findings of this exploratory research conducted in the state, four main findings emerged that will add to the literature on CT integration in elementary science. Interpretations of these findings will be discussed and the policy recommendations. This will be followed by limitations and future research.

Consensus on CT

The computer science and education communities need to come together and decide on a shared terminology of what computational thinking is along with the skills involved in order to have successful implementation. There is vast agreement that all students should know how to think computationally in early education especially with the changing world (Barr and Stephenson 2011; Grover and Pea 2013; ISTE & CSTA, 2011; NGSS, 2013; Wing, 2006). The literature also supports that science is a discipline that integrates well with CT as shown by the taxonomy created by Weintrop et al. (2016). Defining CT for integration at the elementary level is complex especially because there is no shared meaning between different stakeholders (Gane et al., 2021; Kalelioglu, 2018). However, it is necessary to have an agreed-upon, teacher-friendly definition so everyone can use a common language and share resources to have a sustainable implementation process. Creating the definitions needs to be done using consensus with the different stakeholders who are involved and affected by this process.

The research in this dissertation brought to light the value of having professionals from the computer science and education communities come together to decide on a shared terminology of what computational thinking is, along with the skills involved. By having shared language, progress on the different programs can be more accurately assessed. The definitions the DBIR team created are simple and were created by a process called value-mapping, which involved the diverse perspectives of the team. The goal of value-mapping is to create a common language based on multiple points of view (Ryoo and Shea, 2015). The DBIR team brought different

values, experiences, and languages to the table and through collaboration were able to come to a consensus on creating a shared definition. This type of consensus building is important because it also includes the practitioners who will be responsible for the implementation.

Policy makers should consider having teachers meet and discuss definitions for CT concepts and approaches in their practices, like what the DBIR team did, so they can see what skills they use and how they integrate these CT skills into their instruction and share their teachable moments. Ketelhut et al. (2019) contend teachers are able to deepen their science learning when given the time, space, and support to discuss the lesson and their attempts at integrating CT into science activities

Teacher and District Level Factors Impacting CT Integration

According to Bronfenbrenner's Ecological Systems Theory (1979), different layers or systems in a child's environment influence their development and learning. These environments are nested within one another in different layers that interact directly and indirectly. Distinct factors or layers can have an impact on a child's learning of CT such as teacher and district initiatives. Teacher factors and district factors were analyzed using HLM to determine if they had an impact on the teaching and learning of computational thinking skills. Three teacher-level variables were found to have a significant relationship with increasing the frequency of CT concepts and approaches which included grade level taught, teachers' confidence levels, and teacher experience. Professional development frequency and levels of concern were not significant predictors. Grade level and confidence levels were significant

predictors for all concepts and approaches whereas teacher experience was significant for only patterns, abstraction, debugging, and collaboration.

As grade level increases, the frequency of teaching CT concepts and approaches increases. Developmentally students can engage in more complex thinking as they progress through the grades that teachers consider when teaching their lessons (Rijke et al., 2018). Curriculum designers need to consider what is developmentally appropriate when creating CT programs for different grade levels and the level of support needed by teachers from their students.

Teachers with more years of experience engaged in higher frequencies of CT concepts of patterns and abstraction and higher frequencies of approaches of debugging and collaboration. The concepts of recognizing patterns and abstraction involve utilizing higher-order thinking skills when taught. Experienced teachers may be more intentional in promoting higher-order thinking, which successfully advances critical thinking (Miri et al., 2007). Experienced teachers could serve as mentors to novice teachers for helping them integrate CT concepts.

Teachers with higher levels of confidence integrated CT into science instruction more often with all the concepts and approaches. This is a noteworthy finding because districts can provide resources to help increase their staff's confidence levels by implementing professional development that utilizes engaging tools and support structures such as coaching and mentoring (Chalmers, 2018; Israel et al., 2015; Jaipal-Jamani & Angeli, 2017; Ketelhut et al., 2019; Rich et al., 2017). Strategies used during professional development that should be considered are active learning, reflection, collaboration, receiving feedback from coaches and instructors,

and mentoring (Maeng et al., 2020; Nolan and Molla, 2017). Four factors responsible for increasing confidence or self-efficacy are mastery, vicarious experience, social and verbal persuasion, and increasing physical and emotional states (Bandura, 1986). Effective professional development for mastery includes teaching both content and pedagogy. Creating vicarious experiences can happen through collaboration and mentoring. Social and verbal persuasion can improve confidence through a supportive culture and providing positive feedback. Increasing physical states can be achieved through student-centered instruction and increasing positive emotional states through getting teachers to have a shared vision of making a difference in student learning (Bandura, 1986; Lakshmann et al., 2011; Maeng et al., 2020; Sandholtz & Ringstaff, 2014). Teachers are more likely to engage in a practice if they feel like they can be successful at achieving results (Bandura, 1986) so building communities of practice that evaluate instruction and student work and provide informative feedback will also build confidence.

Hours of professional development (PD) were not significant predictors of the CT usage in this research. The current professional development that elementary science teachers are getting is not enough to enhance confidence. This is most likely because more than 70% of the teachers had less than three hours of computer science professional development. Professional development needs to be ongoing, sustained, and coherent to be effective (Maeng et al., 2020). Therefore, professional development should not be a factor that is ruled out when working towards building teachers' confidence. Students will not understand CT until teachers understand and feel confident about their abilities (Israel et al., 2015). More studies are needed to

examine the effects of professional development using strategies that increase teachers' confidence, especially when this study shows the importance of participating in an RPP. PD with more systematic and ongoing support is needed to create a community of practice.

Stages of concern also did not predict CT usage. Most teachers (61%) were only at the beginning stages of concern for implementing CT meaning they may have heard something about it or want to know more about it, but other responsibilities take priority and they have not engaged in the innovation. Teachers increase their concern for an innovation when they engage in the innovation more often (Hall & Hord, 2006; Hall et al., 2011). Stages of concern most likely will change once teachers learn the value of CT and start engaging with it in their teaching and learning. However, the first thing teachers need is to become aware of what CT is and its importance.

The important findings at the district level are commonly known factors such as socioeconomic status and classification of being urban, urban ring, rural or suburban. Important finding about these variables are schools that are in rural or urban locations along with students from lower socioeconomic schools are less likely to have access to computer science opportunities (Code.org, 2021; Kale et al., 2018; Karpinski et al., 2021). However, these factors were not as significant as one would predict. The most important district factor was participation in a research practice partnership (RPP). In other words, having an ongoing well-structured PD at the district level was important to increase the use of CT concepts and approaches.

The research practice partnership that teachers belong to in this study is mandated at the district level and has been supporting science education for 21 years. Teachers receive on-going mandatory professional development on NGSS aligned science instruction. Prior to this study the RPP had not intentionally aimed to integrate CT into the science instruction, beyond what is called for in the NGSS. However, there are benefits of belonging to an RPP because it allows teachers to feel supported while solving problems of practice in their teaching. It helps them become more aware of what the best practices are based on current research and allows them to be more comfortable in taking risks and trying something new by having others to share in their successes and failures which improves teaching and learning (Boulden et al., 2018). This research study has demonstrated that districts using the RPP curriculum may be more apt to integrate CT into their everyday lessons. Being part of an RPP and having mandatory PD allows teachers to have this structure in place where they are given the time and space to discuss what is going well in their instruction.

Schools also should be encouraged to join research practice partnerships with their local universities. Without follow-up support for teachers, the outcomes from professional development decline over time (Sandholtz & Ringstaff, 2020). Being part of a sustained RPP would prevent this decline and provide the support needed as new best practices emerge from research and new concepts and skills are needed as technology advances.

Elementary Science Instructional Time

Analysis of survey data provided information about how CT concepts and approaches are present in a substantial portion of the teaching time spent on science instruction. The average science teaching time was determined to be 151 minutes per week which is higher than the national average but only half the amount of time recommended by the National Science Teachers Association (NSTA) who recommends at least 60 minutes per day (Plumley, 2019; Sparks, 2021). Most time in an elementary school day is devoted to math (321 average minutes/week) and ELA instruction (341 average minutes/week). Each CT concept and approach is present on average more than 40% of the science teaching time. With CT becoming an additional skill we want all of our children to have, and knowing how easily it integrates into science, policymakers need to consider increasing the length of time science is being taught.

However, if science instruction time is increased, policy makers will need to be mindful that only 31% of elementary teachers feel prepared to teach science and most don't feel confident teaching everything they need to cover (Sparks, 2021). In addition, many teachers do not receive the professional development necessary to gain confidence to teach science (McClure et al., 2017). However, CT is a common thread that can be taught in all subjects and is best learned when embedded in class subjects and taught in context using an interdisciplinary approach (Grover, 2018). Teaching CT is not adding one more thing to a teacher's plate but adding value to what teachers are already doing. It will be imperative that teachers engage in effective professional development using strategies that work like using student-centered,

reform-based instruction, and situated learning professional development (Maeng et al., 2020; NRC, 2012; Voogt, 2015). It is important that teachers engage with the CT as a learner and have a growth mindset that CT is not too difficult for them to learn (Estapa et al., 2018).

Opportunities for CT Integration

The first step in implementing an innovation is to build awareness. This research demonstrated that when teachers are presented with opportunities to read and think about CT, they recognize it existing in daily instruction. All CT concepts (abstraction, algorithms, decomposition, and patterns) and approaches (tinkering, debugging, creating, persevering, and collaborating) were present in the different districts that participated in the survey with varying amounts of time, with the average lowest percentage of time being 44% and highest being 67%. In addition, teachers were able to describe opportunities for CT with high accuracy.

Opportunities for teaching CT concepts and approaches are present in curriculum all throughout this northeast state. The next steps would be to explicitly teach the CT concepts and approaches using methods such as framing, prompting, and inviting reflection allowing students and teachers to start sharing a common language with computer science (Rich et al., 2021). Once the school is confident with the “exist” stage they can begin working towards the “enhance” and “extend” in future lessons.

From this analysis, one important policy implication is building awareness needs to start early in all pre-service methods programs. The use of CT modules in the pre-service programs is one method to use that is shown to be effective in getting

pre-service teachers to want to use it in their future classrooms (Yadav et al., 2014). Giving examples of specific resources such as different models and simulations is also effective along with explicitly showing how to teach the different concepts (Adler & Kim, 2018). It is also important to include the pre-service mentor teachers in workshops with the pre-service teachers to familiarize them with CT since it probably is a practice never taught in their teacher certification process (Ketelhut et al., 2019).

In-service teachers also need to build awareness by learning what CT is and what it is not. They next need to learn how to integrate computational thinking into their content areas. This can be done by experts in computer science coaching teachers on ways to integrate CT into what they are already doing in the classroom but done in a way that takes their knowledge, time, and availability into consideration. Teachers will then become aware of where it is already happening in their teaching and make the connection of how they can explicitly teach it. Teachers also need clear examples and resources showing how it can be integrated into curricular areas. Colleges and Universities should continue to collaborate with the computer science community and K–12 educators to lead state professional development programs. These could be done online for convenience to create online communities of practice.

Another policy recommendation is administrators and district leaders need to understand what CT is and why it is so important for students to know CT as a 21st century skills set so professional development is also needed by leaders. Administrators also need to understand how trying new things will impact classroom

management, pedagogy, and evaluations. They need to be opened to having teachers fail at some lessons while they try these innovative approaches.

Limitations and Recommendations for Future Research

Like all research, there are limitations that should be considered as the results from this study are interpreted. There were three major limitations associated with the research in this dissertation. The first limitation is most of the data used in this study was quantitative and based on self-reported survey data. Although a detailed description of each CT concept and approach using definitions, pictures, and examples provided a mutual understanding among teachers, limitations exist because self-reported data is questionable with social desirability. Teachers who participated in taking the survey may have a high level of interest and motivation around the subject matter. Therefore, in the future it is recommended to collect qualitative observational data to examine how teachers integrate CT skill development in their instruction to have a more holistic view.

The second limitation is schools and districts play important roles in teacher practices, but this study did not include relevant district level policy variables. The variables used—school classification and percentage of families below poverty level—were mainly contextual. Participation in an RPP is the only district-level policy variable in this study, and it did not include variables that would capture the quality of their participation. In the future, researchers need to collect relevant policy variables to see the effects of these variables on the implementation of CT at the district level.

The third limitation includes using data from only fifteen out of thirty districts that participated in the survey when running the Hierarchical Linear Modeling (HLM) Analysis due to some districts having fewer than five teachers participating. This changed the sample size from (N=259) to (N=222). Although these data requirements provided more reliable estimates in the HLM analysis, preliminary analysis shows that the excluded districts were mostly those serving lower SES communities. The effects of district RPP participation may be bigger for those districts serving the lower SES communities as it provides critical support for ongoing professional development activities. To investigate the effects of SES and RPP participation clearly and fully, data from these districts need to be collected in the future studies.

Conclusion

Integrating CT into the elementary school day requires time to gather information and data on what the best method is for implementing this change. Information was collected that determined that a shared meaning for CT can be reached and there are many opportunities for CT to be integrated into science instruction with specific examples of what this looks like. This research adds to the literature in verifying science is an ideal subject to integrate CT for which all students have access. This research also adds to the literature in determining factors that affect the use of CT which include grade level, experience, confidence, and belonging to an RPP. This work provides the necessary foundation for future implementation efforts. From this work future strategies for implementation need to involve creating a state-wide vision with clear goals where all stakeholders are on board. A systematic roll-out needs to be considered for teacher professional development and suitable

assessments. Lastly, integrating CT needs to have continued support so it can be sustainable along with effective communication between all the people involved. This research will be helpful for curriculum designers, policy makers, teachers, and district leaders by laying the foundation for an essential component needed in achieving computer science for all.

BIBLIOGRAPHY

- 2018 State of Computer Science Education. (2018). Retrieved from <https://advocacy.code.org/>
- Adler, R. F., & Kim, H. (2018). Enhancing future K-8 teachers' computational thinking skills through modeling and simulations. *Education and Information Technologies*, 23(4), 1501-1514. <https://doi.org/10.1007/s10639-017-9675-1>
- Aho, A. V. (2012). Computation and Computational Thinking. *Computer Journal*, 55(7), 832-835. <https://doi.org/10.1093/comjnl/bxs074>
- 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. *Journal of Educational Technology & Society*, 19(3).
- Angeli, C., & Jaipal-Jamani, K. (2018). Preparing pre-service teachers to promote computational thinking in school classrooms. In *Computational thinking in the stem disciplines: Foundations and research highlights* (pp. 127-150). Springer, International Publishing.
- Baek, Y., Wang, S., Yang, D., Ching, Y. H., Swanson, S., & Chittoori, B. (2019). Revisiting second graders' robotics with an Understand/Use-Modify-Create (U2MC) strategy. *European Journal of STEM Education*, 4(1), 1-12. <https://doi.org/10.20897/ejsteme/5772>
- Bandura, A. (1986). Social foundations of thought and action: A social cognitive theory. (pp. 5-107). Prentice Hall.
- Banilower, E. R., Smith, P. S., Malzahn, K. A., Plumley, C. L., Gordon, E. M., & Hayes, M. L. (2018). Report of the 2018 NSSME+. *Horizon Research, Inc.*
- Barefoot Computing. (n.d.). Retrieved from <https://www.barefootcomputing.org>
- 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/10.1145/2676723.2693616>
- Bart, A. C. (2015, February). Situating computational thinking with big data: Pedagogy and technology. In *Proceedings of the 46th ACM technical symposium on computer science education* (pp. 719-719).
- Berland, M., & Wilensky, U. (2015). Comparing Virtual and Physical Robotics Environments for Supporting Complex Systems and Computational Thinking. *Journal of Science Education and Technology*, 24(5), 628-647. <https://doi.org/10.1007/s10956-015-9552-x>
- Berry, M. (2013) Computing in the National Curriculum: A Guide for Primary Teachers, Computing at School, Bedford.

- Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in the early childhood classroom*. Routledge.
- Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education*, 72, 145-157.
<https://doi.org/10.1016/j.compedu.2013.10.020>
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education-Implications for policy and practice* (No. JRC104188). Joint Research Centre (Seville site).
- Bocconi, S., Chiocciariello, A., & Earp, J. (2018). The Nordic approach to introducing Computational Thinking and programming in compulsory education. *Report prepared for the Nordic@ BETT2018 Steering Group*, 397-400.
- Boulden, D. C., Wiebe, E., Akram, B., Aksit, O., Buffum, P. S., Mott, B., Boyer, K.E., & Lester, J. (2018). Computational thinking integration into middle grades science classrooms: Strategies for meeting the challenges. *Middle Grades Review*, 4(3) 1-17.
<https://scholarworks.uvm.edu/mgreview/vol4/iss3/5>
- Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017, November). Development of computational thinking skills through unplugged activities in primary school. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 65-72).
<https://doiorg.uri.idm.oclc.org/10.1145/3137065.3137069>
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada* (Vol. 1, p. 25).
- Breslyn, W., & McGinnis, J. R. (2019). Investigating preservice elementary science teachers' understanding of climate change from a computational thinking systems perspective. *Eurasia Journal of Mathematics, Science and Technology Education*, 15(6) doi:10.29333/ejmste/103566
- Bronfenbrenner, U. (1996). *The ecology of human development experiments by nature and design*. Harvard University Press.
- Bureau of Labor Statistics (2021). U.S. Department of Labor, Computer and Information Research Scientists: *Occupational Outlook Handbook*. Retrieved from <https://www.bls.gov/ooh/computer-and-information-technology/computer-and-information-research-scientists.htm>
- Butler, D., & Leahy, M. (2021). Developing preservice teachers' understanding of computational thinking: A constructionist approach. *British Journal of*

- Educational Technology*, 52(3), 1060–1077.
<https://doi.org/10.1111/bjet.13090>
- Bybee, R. W. (2013). *The case for STEM education: Challenges and opportunities*. NSTA Press.
- Cateté, V., Lytle, N., Dong, Y., Boulden, D., Akram, B., Houchins, J., Barnes, T., Wiebe, E., Lester, J., Mott, B. & Boyer, K. (2018). Infusing computational thinking into middle grade science classrooms: lessons learned. In *Proceedings of the 13th Workshop in Primary and Secondary Computing Education* (pp. 1-6).
- CAST (2018). Universal Design for Learning Guidelines version 2.2. Retrieved from <http://udlguidelines.cast.org>
- Chalmers, C. (2018). Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17, 93-100.
<https://doi.org/10.1016/j.ijcci.2018.06.005>
- Coburn, C. E., & Penuel, W. R. (2016). Research–practice partnerships in education: Outcomes, dynamics, and open questions. *Educational Researcher*, 45(1), 48-54. <https://doi.org/10.3102/0013189X16631750>
- Coburn, C. E., Penuel, W. R., & Farrell, C. C. (2021). Fostering educational improvement with research-practice partnerships. *Phi Delta Kappan*, 102(7), 14-19. <https://doi.org/10.1177/00317217211007332>
- Code.org, CSTA, & ECEP Alliance. (2020). 2020 State of Computer Science Education: Illuminating Disparities. Retrieved from <https://advocacy.code.org/stateofcs>
- Code.org, CSTA, & ECEP Alliance. (2021). 2021 State of computer science education: Accelerating action through advocacy. Retrieved from <https://advocacy.code.org/stateofcs>
- Computer Science Teachers Association (2020). *Standards for Computer Science Teachers*. Retrieved from <https://csteachers.org/teacherstandards>
- Computational Thinking Curriculum, Computer Science Lessons. (n.d.). Retrieved from <https://nicerc.org/curricula/computer-science/>
- Computational Thinking for Educators—Unit 1: Introducing Computational Thinking. (2015). Retrieved from <https://computationalthinkingcourse.withgoogle.com/unit>
- Computing At School. (2013). Computing in the national curriculum: A guide for primary teachers. Belford, UK: Newnorth Print. Retrieved from <http://www.computingatschool.org.ukdatauploads/CASPrimaryComputing>

- Count, R. I. (2018). Policy & Advocacy for Rhode Island's Children. Retrieved from <http://www.rikidscount.org/>
- CS for All (2022). Computer Science for All. January 1, 2022 from <https://www.csforall.org>
- CS4RI (2022). Retrieved January 1, 2022, from <https://www.cs4ri.org/ri-cs-education-standards>
- Creswell, J. W. (2014). The selection of a research approach. *Research design: Qualitative, quantitative, and mixed methods approaches*, 3-24
- CS for All (2021). Computer Science for All. Retrieved October 26, 2021, from <https://www.csforall.org>
- Curran, F. C., & Kitchin, J. (2019). Early elementary science instruction: Does more time on science or science topics/skills predict science achievement in the early grades? *AERA Open*, 5(3), 2332858419861081.
- Cuny, J., Snyder, L., Wing, J.M. (2010). Demystifying computational thinking for non-computer scientists. Unpublished manuscript, referenced in <http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf>.
- Dasgupta, A., Rynearson, A. M., Purzer, S., Ehsan, H. and Cardella, M. E. (2017). Computational thinking in K-2 Classrooms: Evidence from student artifacts (Fundamental). A paper presented at American Society for Engineering Education Annual Conference and Exhibition, June 25-28, 2017, Columbus, Ohio. <https://doi.org/10.18260/1-2--28062>
- Davis, E. & Smithey, J. (2009). Beginning teachers moving toward effective elementary science teaching. *Science Education (Salem, Mass.)*, 93(4), 745–770. <https://doi.org/10.1002/sce.20311>
- Denning, P. J., & Tedre, M. (2019). *Computational Thinking*. MIT Press.
- Dillman, D. A., Smyth, J. D., & Christian, L. M. (2014). *Internet, phone, mail, and mixed-mode surveys: The tailored design method*. John Wiley & Sons.
- Dong, Y., Catete, V., Jocius, R., Lytle, N., Barnes, T., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2019, February). PRADA: A practical model for integrating Computational thinking in K–12 education. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 906–912). <https://doi.org/10.1145/3287324.3287431>
- Ehsan, H., Rehmat, A. P., & Cardella, M. E. (2021). Computational thinking embedded in engineering design: capturing computational thinking of children in an informal engineering design activity. *International Journal of*

- Technology & Design Education*, 31(3), 441–464. <https://doi-org.uri.idm.oclc.org/10.1007/s10798-020-09562-5>
- Every Student Succeeds Act, 20 U.S.C. § 6301 (2015).
<https://www.congress.gov/114/plaws/publ95/PLAW-114publ95.pdf>
- Fisher, L. M. (2016). A decade of ACM efforts contribute to computer science for all. *Communications of the ACM*, 59(4), 25–27. <https://doi.org/10.1145/2892740>.
- Gane, B. D., Israel, M., Elagha, N., Yan, W., Luo, F., & Pellegrino, J. W. (2021). Design and validation of learning trajectory-based assessments for computational thinking in upper elementary grades. *Computer Science Education*, 31(2), 141–168. <https://doi-org.uri.idm.oclc.org/10.1080/08993408.2021.1874221>
- GEMS-Net (2020). GEMSNET Guiding Education in Math and Science Network. Retrieved from <https://web.uri.edu/gemsnet/>
- Google Inc. (2015). Searching for Computer Science: Access and Barriers in U.S. K–12 Education. Retrieved from https://services.google.com/fh/files/misc/searching-for-computer-science_report.pdf
- Google Inc. & Gallup Inc. (2017, December). Encouraging Students Toward Computer Science Learning. Results From the 2015-2016 Google-Gallup Study of Computer Science in the U.S. K–12 Schools (Issue Brief No. 5). Retrieved from <https://goo.gl/iM5g3A>.
- Grover, S. (2018). The 5th ‘C’ of 21st century skills. *Try computational thinking (not coding)*. (March 13). Retrieved from EdSurge News: <https://www.edsurge.com/news/2018-02-25-the-5th-c-of-21st-century-skills-try-computational-thinking-not-coding>.
- Grover, S., & Pea, R. (2018). Computational thinking: a competency whose time has come. In S. Sentance, E. Barendsen, & S. Carsten (Eds.), *Computer science education: perspectives on teaching and learning in school* (pp. 19–37). London: Bloomsbury Academic.
- Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, 42(1), 38-43.
- Guba, E. G. & Lincoln, Y.S. (1994). Competing paradigms in qualitative research. In N. Denzin & Lincoln, Y.S., *Handbook of qualitative research* (1st Ed. 1994). (pp.105-117). Thousand Oaks, CA: Sage Publications, Inc.
- Hall, G. E., & Hord, S. M. (2006). Implementing change: Patterns, principles, and potholes. Pearson.

- Hall, G. E., Hord, S. M., Aguilera, R., Zepeda, O., & von Frank, V. (2011). Implementation: Learning builds the bridge between research and practice. *The Learning Professional*, 32(4), 52-57.
- Henderson, P. B., Cortina, T. J., & Wing, J. M. (2007). Computational thinking. *ACM SIGCSE Bulletin*, 39(1), 195-196.
- Hestness, E., Ketelhut, D. J., McGinnis, J. R., & Plane, J. (2018). Professional knowledge building within an elementary teacher professional development experience on computational thinking in science education. *Journal of Technology and Teacher Education*, 26(3), 411-435. <https://www.learntechlib.org/primary/p/181431/>
- Hord, S. M., & Roussin, J. L. (2013). *Implementing change through learning: Concerns-based concepts, tools, and strategies for guiding change*. Corwin Press.
- Hsu, Irie, N. R., & Ching, Y.-H. (2019). Computational Thinking Educational Policy Initiatives (CTEPI) Across the Globe. *Tech Trends*, 63(3), 260–270. <https://doi.org/10.1007/s11528-019-00384-4>
- Hsu, Y. S. (2015). The development of teachers' professional learning and knowledge. *Development of science teachers' TPACK*, 3-15.
- Instruction & Assessment. RI Model Science Curriculum. (n.d.). Retrieved January 19, 2022, from <https://www.ride.ri.gov/InstructionAssessment/Science/>
- Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, 82, 263-279. <https://doi.org/10.1016/j.compedu.2014.11.022>
- Iyer, S. (2019). Teaching-Learning of Computational Thinking in K–12 Schools in India. In *Computational Thinking Education* (pp. 363-382). Springer.
- ISTE, C. (2011). Computational Thinking in K–12 Education leadership toolkit. *Computer Science Teacher Association*: <http://csta.acm.org/Curriculum>
- ISTE, CSTA. (2011). *Computational thinking in K–12 education leadership toolkit*.
- ISTE (2018). ISTE Standards for Educators: Computational Thinking Competencies. (International Society for Technology in Education). <https://www.iste.org>.
- Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, 26(2), 175–192. <https://doi-org.uri.idm.oclc.org/10.1007/s10956-016-9663-z>

- K–12 Computer Science Framework (2016). *K–12 computer science framework*. <https://www.k12cs.org/>
- Kafura, D., Bart, A. C., & Chowdhury, B. (2018). A computational thinking course accessible to non-stem majors. *Journal of Computing Sciences in Colleges*, 34(2), 157-163.
- Kale, U., Akcaoglu, M., Cullen, T., & Goh, D. (2018). Contextual factors influencing access to teaching computational thinking. *Computers in the Schools*, 35(2), 69-87. <https://doi-org.uri.idm.oclc.org/10.1080/07380569.2018.1462630>
- Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 4(3), 583-596.
- Kang, E. J. S., Donovan, C., & McCarthy, M. J. (2018). Exploring Elementary Teachers' Pedagogical Content Knowledge and Confidence in Implementing the NGSS Science and Engineering Practices. *Journal of Science Teacher Education*, 29(1), 9–29. <https://doi-org.uri.idm.oclc.org/10.1080/1046560X.2017.1415616>
- Karpinski, Z., Biagi, F., & Di Pietro, G. (2021). Computational Thinking, Socioeconomic Gaps, and Policy Implications. IEA Compass: Briefs in Education. Number 12. *International Association for the Evaluation of Educational Achievement*.
- Kaya, E., Newley, A., Yesilyurt, E., & Deniz, H. (2020). Measuring Computational Thinking Teaching Efficacy Beliefs of Preservice Elementary Teachers. *Journal of College Science Teaching*, 49(6), 55–64.
- 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*, 1-15. <https://doi.org/10.1007/s10956-019-09798-4>
- Khine, M. S. (Ed.). (2018). *Computational Thinking in the STEM Disciplines: Foundations and Research Highlights*. Springer.
- Koehler, M., & Mishra, P. (2009). What is technological pedagogical content knowledge (TPACK)? *Contemporary Issues in Technology and Teacher Education*, 9(1), 60-70.
- Laerd Statistics (2018). Spearman's correlation using SPSS Statistics. *Statistical tutorials and software guides*. Retrieved from <https://statistics.laerd.com/>
- Lakshmanan, A., Heath, B., Perlmutter, A., & Elder, M. (2011). The impact of science content and professional learning communities on science teaching

- efficacy and standards-based instruction. *Journal of Research in Science Teaching*, 48, 534–551.
- Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2019). Computational thinking from a disciplinary perspective: Integrating computational thinking in K–12 Science, Technology, Engineering, and Mathematics education. *Journal of Science Education and Technology*, 29(1), 1–8. (2020) 29:1–8. <https://doi.org/10.1007/s10956-019-09803-w>
- Lee, I., & Malyn-Smith, J. (2020). Computational Thinking Integration Patterns Along the Framework Defining Computational Thinking from a Disciplinary Perspective. *Journal of Science Education and Technology*, 29(1), 9–18. <https://doi.org/10.1007/s10956-019-09802-x>
- Lister, R. (2011). Concrete and other neo-piagetian forms of reasoning in the novice programmer. *Conferences in Research and Practice in Information Technology Series*, 114, 9–18.
- Maeng, Whitworth, B. A., Bell, R. L., & Sterling, D. R. (2020). The effect of professional development on elementary science teachers' understanding, confidence, and classroom implementation of reform-based science instruction. *Science Education (Salem, Mass.)*, 104(2), 326–353. <https://doi.org/10.1002/sce.21562>
- Maltese, A. V., & Tai, R. H. (2010). Eyeballs in the fridge: Sources of early interest in science. *International Journal of Science Education*, 32(5), 669–685. <https://doi-org.uri.idm.oclc.org/10.1080/09500690902792385>
- Martínez-García, E. (2021). Computational thinking: the road to success in education. *Academian Letters*, Article 3973. <https://doi.org/10.20935/AL3973>.
- Mattera, S., & Morris, P. (2017). *Counting on Early Math Skills: Preliminary Kindergarten Impacts of the Making Pre-K Count and High 5s Programs*. MDRC.
- Massachusetts Department of Elementary and Secondary Education. (2016). *Massachusetts Digital Literacy and Computer Science Curriculum Framework*. Retrieved October 31, 2021, from <http://www.doe.mass.edu/stem/dlcs/>
- McClure, E. R., L. Guernsey, D. H. Clements, S. N. Bales, J. Nichols, N. Kendall-Taylor, and M. H. Levine. 2017. *STEM starts early: Grounding science, technology, engineering, and math education in early childhood*. New York: The Joan Ganz Cooney Center at Sesame Workshop.
- Merriam-Webster. (n.d.) Merriam-Webster.com dictionary. Retrieved November 26, 2021, From <https://www.merriam-webster.com/>

- McGinnis, J. R., Ketelhut, D. J., Mills, K., Hestness, E., Jeong, H., & Cabrera, L. (2019). Preservice Science Teachers' Intentions and Avoidances to Integrate Computational Thinking into Their Science Lesson Plans for Young Learners. *Grantee Submission*.
- Mcintosh, K., Horner, R., Chard, D., Boland, J., & Good, R. (2006). The use of reading and behavior screening measures to predict nonresponse to school-wide positive behavior support: A longitudinal analysis. *School Psychology Review*, 35(2), 275-291. <https://doi-org.uri.idm.oclc.org/10.1080/02796015.2006.12087992>
- Metz, S. S. (2007). Attracting the engineers of 2020 today. *Women and minorities in science, technology, engineering, and mathematics: Upping the numbers*, 184-209.
- Mills, K., Coenraad, M., Ruiz, P., Burke, Q., & Weisgrau J. (2021, December). Computational thinking for an inclusive world: A resource for educators to learn and lead. Digital Promise. <https://doi.org/10.51388/20.500.12265/138>
- Miri, B., David, B. C., & Uri, Z. (2007). Purposely teaching for the promotion of higher-order thinking skills: A case of critical thinking. *Research in science education*, 37(4), 353-369. <https://doi.org/10.1007/s11165-006-9029-2>
- Montoya. (2017). Computer Science for All: Opportunities Through a Diverse Teaching Workforce. *Harvard Journal of Hispanic Policy*, 29.
- Mulvey, K. L., McGuire, L., Hoffman, A. J., Hartstone-Rose, A., Winterbottom, M., Balkwill, F., Fields, G. E., Burns, K., Drews, M., Chatton, M., Eaves, N., Law, F., Joy, A., & Rutland, A. (2020). Learning hand in hand: Engaging in research-practice partnerships to advance developmental science. *New Directions for Child and Adolescent Development*, 2020 (172), 125-134. <https://doi-org.uri.idm.oclc.org/10.1002/cad.20364>
- National Center for Education Statistics (NCES) U.S. Department of Education. (n.d.). Retrieved from <https://nces.ed.gov/>
- National Research Council. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press.
- NGSS Lead States. (2013). *Next generation science standards: For states, by states*. Washington, D.C.: National Academies Press.
- Nolan, & Molla, T. (2017). Teacher confidence and professional capital. *Teaching and Teacher Education*, 62, 10-18. <https://doi.org/10.1016/j.tate.2016.11.004>
- Noonan, R., & United States. Economics Statistics Administration, issuing body. (2017). *STEM Jobs: 2017 Update* (ESA issue brief; 17-02). Washington, DC:

U.S. Department of Commerce, Economics and Statistics Administration,
Office of the Chief Economist.

NSTA. (n.d.). *About the next generation science standards*. NGSS@NSTA. Retrieved December 26, 2021, from <https://ngss.nsta.org/about.aspx>

Ogegbo, A. A., & Ramnarain, U. (2021). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, 1–28.
<https://doi-org.uri.idm.oclc.org/10.1080/03057267.2021.1963580>

Ouyang, Y., Hayden, K. L., & Remold, J. (2018, February). Introducing Computational Thinking through Non-Programming Science Activities. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 308-313). ACM. <https://doi.org/10.1145/3159450.3159520>

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

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, 1(1), 95-123.

Patton, M. Q. (2015). *Qualitative research & evaluation methods: integrating theory and practice*. Thousand Oaks, CA: SAGE Publications, Inc.

Peel, A., Sadler, T. D., & Friedrichsen, P. (2019). Learning natural selection through computational thinking: Unplugged design of algorithmic explanations. *Journal of Research in Science Teaching*, 56(7), 983-1007.
<https://doi.org/10.1002/tea.21545>

Penuel, W. R., Coburn, C. E., & Gallagher, D. J. (2013). Negotiating problems of practice in research–practice design partnerships. *National Society for the Study of Education Yearbook*, 112(2), 237-255.

Penuel, W. R., Fishman, B. J., Yamaguchi, R., & Gallagher, L. P. (2007). What makes professional development effective? Strategies that foster curriculum implementation. *American Educational Research Journal*, 44(4), 921-958.
<https://doi.org/10.3102/0002831207308221>

Pietros, J., Shim, M., Sweetman, S., Hamouda, S., & Goodrow, A. (2022). *Predicting Computational Thinking in Elementary Science using a Multilevel Model Approach* [Manuscript submitted for publication] Education, University of Rhode Island

Pietros, J. & Sweetman, S. (2020). The importance of research practice partnerships for professional development. *Journal of the National Association for Professional Development Schools*, Special Edition Partners: Bridging Research to Practice, 15(2) 23-25.

- Plumley, C. L. (2019). 2018 NSSME+: Status of elementary school science. Chapel Hill, NC: Horizon Research, Inc.
- Prottzman, K. (2019). *Computational Thinking Meets Student Learning: extending the iste standards*. International Society for Technology in Education.
- Raudenbush, S. W., & Bryk, A. S. (2002). *Hierarchical Linear Models: Applications and Data Analysis Methods* (2nd Edition). Thousand Oaks, CA: Sage Publications, Inc.
- Rhode Island Computer Science Education Standards. (2018)
<https://www.cs4ri.org/ri-cs-education-standards>
- Rhode Island K–12 Computer Science Education Standards. (2018). *Rhode Island K-12 Computer Science Education Standards*. Retrieved November 27, 2021, From https://www.ride.ri.gov/Portals/0/Uploads/Documents/Instruction-and-Assessment-World-Class-Standards/Other-Subjects/RI_CS_Ed_Standards
- Rich, Egan, G., & Ellsworth, J. (2019). A Framework for Decomposition in Computational Thinking. *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, 416–421.
<https://doi.org/10.1145/3304221.3319793>
- Rich, K. M., Yadav, A., & Schwarz, C. V. (2019). Computational Thinking, Mathematics, and Science: Elementary Teachers' Perspectives on Integration. *Journal of Technology & Teacher Education*, 27(4), 165–205
- Rich, K. M., Yadav, A., & Larimore, R. A. (2020). Teacher Implementation Profiles for Integrating Computational Thinking into Elementary Mathematics and Science Instruction. *Education and Information Technologies*, 25(4), 3161–3188. <https://doi.org/10.1007/s10639-020-10115-5>
- Rich, P. J., Jones, B., Belikov, O., Yoshikawa, E., & Perkins, M. (2017). Computing and engineering in elementary school: The effect of year-long training on elementary teacher self-efficacy and beliefs about teaching computing and engineering. *International Journal of Computer Science Education in Schools*, 1(1), 1-20. <https://doi.org/10.21585/ijcses.v1i1.6>
- Rhode Island Board of Education Act, RIGL § 16-22- 31(2021).
<https://law.justia.com/codes/rhode-island/2021/title-16/chapter-16-22/section-16-22-31/>
- Rijke, W.J., Bollen, L., Eysink, T. H. S., & Tolboom, J. (2018). Computational Thinking in Primary School: An Examination of Abstraction and Decomposition in Different Age Groups. *Informatics in Education*, 17(1), 77–92. <https://doi.org/10.15388/infedu.2018.05>

- Rischar, J. F. (2007). *High noon: 20 global problems, 20 years to solve them*. Hachette UK.
- Ruberg, L. F., & Owens, A. (2017). A future-focused education: Designed to create the innovators of tomorrow. In *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 367-392). Springer, Cham.
- Ryoo, J & Shea, M. (2015) Value mapping: An activity for surfacing power dynamics and diverse perspectives in research-practice collaborations. *San Francisco, CA: Exploratorium*.
- Sandholtz, J. H., & Ringstaff, C. (2014). Inspiring instructional change in elementary science: The relationship between enhanced self-efficacy and teacher practices. *Journal of Science Teacher Education*, 25, 729–751.
- Sandholtz, J. H., & Ringstaff, C. (2020). Offering modest supports to extend professional development outcomes and enhance elementary science teaching. *Professional Development in Education*, 1-16. <https://doi.org/10.1080/19415257.2020.1725594>
- Sands, P., Yadav, A., & Good, J. (2018). Computational thinking in K–12: In-service teacher perceptions of computational thinking. In *Computational thinking in the STEM disciplines* (pp. 151-164). Springer, Cham.
- Sengupta, P., Kinnebrew, J., Basu, S., Biswas, S., & Clark, G. (2013). Integrating Computational thinking with K–12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18(2), 351-380. <https://doi.org/10.1007/s10639-012-9240-x>
- 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>
- Sparks, S. D. (2021, July 14). Make science education better, more equitable, says National Panel. *Education Week*. Retrieved January 8, 2022, from <https://www.edweek.org/teaching-learning/make-science-education-better-more-equitable-says-national-panel/2021/07>
- Sweetman, S. (Principal Investigator). (2018-2021). *Computing in Elementary School: An Exploration of Computational Thinking Approaches and Concepts Across Disciplines* (Award number: 1813224) [Grant] National Science Foundation.
- Teachers & Administrators. (n.d.). Retrieved from <https://www.ride.ri.gov/TeachersAdministrators/EducatorCertification.aspx>.
- Usengül, L., & Bahçeci, F. (2020). The effect of LEGO WeDo 2.0 Education on academic achievement and attitudes and computational thinking skills of

- learners toward science. *World Journal of Education*, 10(4), 83- 93.
<https://doi.org/10.5430/wje.v10n4p83>
- Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 20(4), 715-728.
- 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(1), 53-64.
<https://doi.org/10.1007/s10956-019-09801-y>
- Webb, M., Davis, N., Bell, T., Katz, Y. J., Reynolds, N., Chambers, D. P., & Syslo, M. (2017). Computer science in K–12 school curricula of the 21st century: Why, what, and when? *Education and Information Technologies*, 22(2), 445-468.
<https://doi.org/10.1007/s10639-016-9493-x>
- Weinberg, A. E. (2013). *Computational thinking: An investigation of the existing scholarship and research* (Doctoral dissertation, Colorado State University).
- Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1), 127-147.
<https://doi.org/10.1007/s10956-015-9581-5>
- White, E., & Shakibnia, A. F. (2019). State of STEM: Defining the Landscape to Determine High-Impact Pathways for the Future Workforce. In *Proceedings of the Interdisciplinary STEM Teaching and Learning Conference* (Vol. 3, No. 1, p. 4). DOI: 10.20429/stem.2019.030104
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35. <https://doi.org/10.1145/1118178.1118215>
- Wing, J. (2011). Research notebook: Computational thinking—What and why. *The link magazine*, 6, 20-23.
- Woltman, H., Feldstain, A., MacKay, J. C., & Rocchi, M. (2012). An introduction to hierarchical linear modeling. *Tutorials in quantitative methods for psychology*, 8(1), 52-69.
- Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In *Emerging research, practice, and policy on computational thinking* (pp. 205-220). Springer, Cham.
- Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K–12

classrooms. *TechTrends*, 60(6), 565–568. <https://doi.org/10.1007/s11528-016-0087-7>

Yadav, A., Larimore, R., Rich, K., & Schwarz, C. (2019, March). Integrating computational thinking in elementary classrooms: Introducing a toolkit to support teachers. In *Society for Information Technology & Teacher Education International Conference* (pp. 347-350). Association for the Advancement of Computing in Education (AACE).

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, 14(1), 5, 1-16. <https://doi.org/10.1145/2576872>

Zhao, M., Zhao, M., Wang, X. H., & Ma, H. L. (2020, December). Promoting Teaching Self-efficacy in Computational Thinking of School Teachers. In *2020 Ninth International Conference of Educational Innovation through Technology (EITT)* (pp. 235-239). IEEE. doi:10.1109/EITT50754.2020.00048

Appendix A: Computational Thinking in Elementary School Classrooms

Welcome to the Computational Thinking Survey

This survey is being conducted to understand the concepts and approaches to computational-thinking being used in K–5 Schools. Your participation is voluntary. No personally identifiable information will be associated with your responses in any reports of the data. Upon completion of this survey, a form from URI will be available to print out for 1 PLU (Professional Learning Unit) credit. If you have any questions or comments about the survey, please feel free to contact Sara Sweetman, the study director, by email at sara_sweetman@uri.edu or by phone (401) 874-600

1. Do you agree to participate in the survey?

- ☐ Yes, I agree to participate
- ☐ No, I do not agree to participate

Computational Thinking in Elementary School Classrooms

2. Which statement best describes your concern about teaching computational thinking curriculum in elementary school?

- ☐ "I've heard something about it, but other responsibilities take priority"
- ☐ "This seems interesting, and I would like to know more about it."
- ☐ "I'm concerned about the changes I will need to make in my routines."
- ☐ "I'm concerned about how much time it takes to get ready to teach with this new approach."
- ☐ "How will this new approach impact my students?"
- ☐ "I'm looking forward to sharing some ideas about it with other teachers."
- ☐ "I incorporate computational thinking skills into my lessons now and have ideas about how to do it better."

3. How would you rate your confidence level in teaching computational thinking?

- ☐ Extremely confident
- ☐ Very confident
- ☐ Somewhat confident
- ☐ Not so confident
- ☐ Not at all confident

4. How would you define your gender?

- ☐ Female
- ☐ Male
- ☐ Non-binary
- ☐ Prefer to self-describe

5. Which race/ethnicity best describes you? (Please choose only one.)

- ☐ American Indian or Alaskan Native
- ☐ Asian / Pacific Islander
- ☐ Black or African American
- ☐ Hispanic
- ☐ White / Caucasian
- ☐ Multiple ethnicity / Other (please specify)

6. In which school district or educational organization are you employed or are you representing?

7. What is the name of your school?

8. How long have you been an educator?

- ☐ 0-3 years
- ☐ 4-6 years
- ☐ 7-10 years
- ☐ 11-15 years
- ☐ 16-20 years
- ☐ 20+ years

9. How long have you been in your current position?

- ☐ 0-3 years
- ☐ 4-6 years
- ☐ 7-10 years
- ☐ 11-15 years
- ☐ 16-20 years
- ☐ 20+ years

10. What areas are you certified?

11. What is the highest level of education you have completed?

- ☐ Bachelor's Degree
- ☐ Some graduate school
- ☐ Master's Degree
- ☐ Master's Degree plus additional credits
- ☐ PhD / EdD

12. What grade(s) do you teach? Select all that apply.

- ☐ Kindergarten
- ☐ First Grade
- ☐ Second Grade
- ☐ Third Grade
- ☐ Fourth Grade
- ☐ Fifth Grade
- ☐ Other (please specify)

13. What subjects do you teach? Select all that apply.

- ☐ Math
- ☐ Science
- ☐ Social Studies
- ☐ English or Language Arts
- ☐ After school computer programming
- ☐ Other specialization

Other specialization (please specify)

* 14. On average how many days a week are your students taught each of the following subjects?

	0	1	2	3	4	5
ELA	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Math	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Science	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Social Studies	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Specialization	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

* 15. On average how many minutes is a typical lesson in each of the following subjects?

	0-20 min	21-40 min	41-60 min	61-80 min	81-100 minutes	greater than 100 minutes
ELA	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Math	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Science	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Social Studies	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Specialization	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

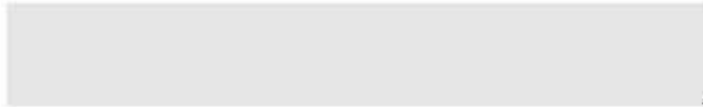
16. What type of science curriculum are you using? (check all that apply)

- ☐ Stemsscopes
- ☐ Gizmos
- ☐ Gems-Net/Foss
- ☐ Kit based curriculum
- ☐ OpenSciEd
- ☐ Make my own lessons
- ☐ Other (please specify)

17. How often do you use technology for instruction in your classroom?

- ☐ Most lessons each day
- ☐ A few lessons each day
- ☐ One lesson a day
- ☐ A few lessons each week
- ☐ A few lessons each month
- ☐ A few lessons each year
- ☐ Never

18. Briefly describe or list the types of technology you or your students are using.



Computational Thinking in Elementary School Classrooms

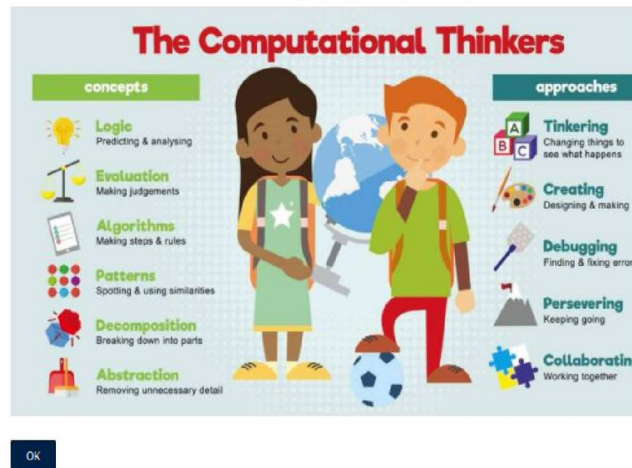
19. Computational thinking (CT) is a problem solving process that includes a number of characteristics and dispositions. CT is essential to the development of computer applications, but it can also be used to support problem-solving across all disciplines, including the humanities, math, and science. Students who learn CT across the curriculum can begin to see a relationship between academic subjects, as well as between life inside and outside of the classroom ("Google Computational Thinking for Educators", n.d.)

Computational thinking allows us to take a complex problem, understand what the problem is and develop possible solutions. We can then present these solutions in a way that a computer, a human, or both, can understand. (<https://www.bbc.com/bitesize/guides/zp92mp3/revision/1>)

Based on these definitions, how often do you think you apply computational thinking in your classroom?

- ☐ Most lessons each day
- ☐ A few lessons each day
- ☐ One lesson a day
- ☐ A few lessons each week
- ☐ A few lessons each month
- ☐ A few lessons each year
- ☐ Never

For each of the computational thinking concepts and approaches listed in the picture below, we describe the concept or approach and then ask you to share how often you teach or use the concept or approach in each subject. Finally we ask for an example of how you might teach this in the classroom. You do not have to give an example for each concept and approach. It is our hope that out of the different concepts and approaches you pick just a few to share a short (a few sentences) example description with us.



Computational Thinking in Elementary School Classrooms

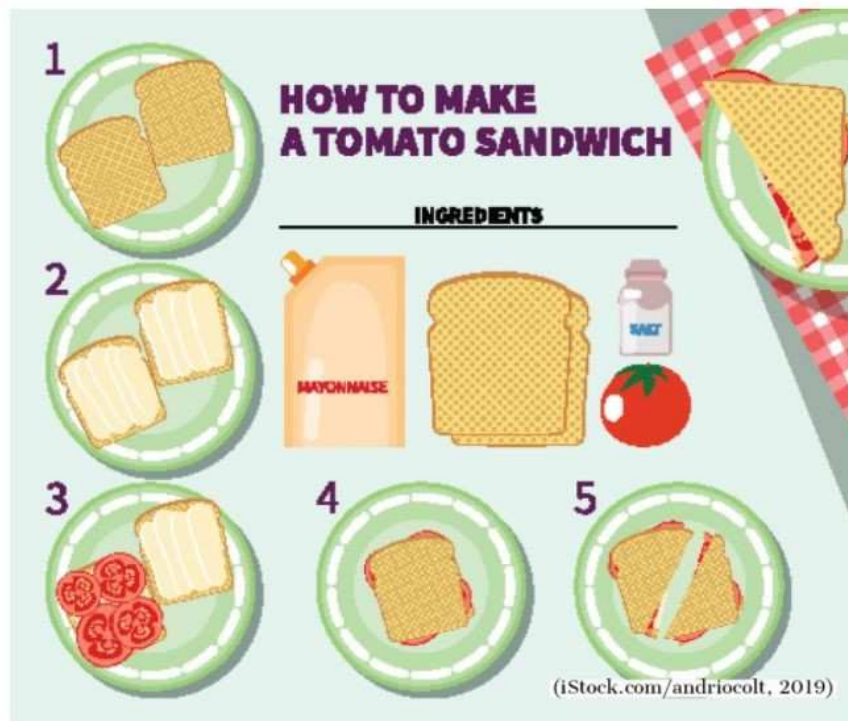
Concept: Algorithms

An algorithm is "a sequence of instructions or a set of rules to get something done. Computer scientists strive for algorithms which solve problems in the most-effective and efficient ways - getting the most-accurate results, in the quickest time, with the fewest resources (memory or time)" (Barefoot Computing, n.d.).

When students come up with their own sequences of instructions, for example, how to get dressed or clean their teeth, create a plan for a story, write instructions for a game or sport, develop rules for grammar or math, they are creating algorithms (www.barefootcas.org.uk).

OK

This is an example of an algorithm for making a tomato sandwich.



* 20. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in algorithms?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

* 21. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching algorithms?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

22. In a sentence or two describe a lesson you taught that includes **algorithms** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

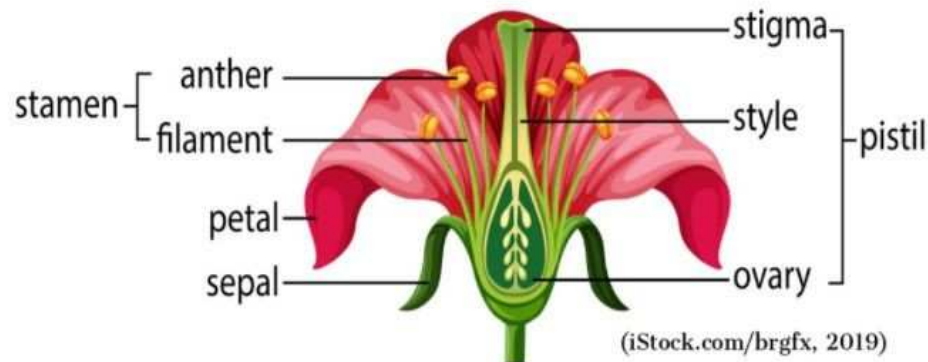
Concept: Decomposition

"Decomposition is the process of breaking down a task into smaller, more-manageable parts. It has many advantages. It helps us manage large projects and makes the process of solving a complex problem less daunting and much easier to take on" (Barefoot Computing, n.d.). Whenever students are labelling, adding detail to concept maps, or creating instructions, life cycles, and timelines, they are practicing their decomposition skills. Solving a math problem, getting dressed for PE, planning a research project, or organizing a school event are other examples. (www.barefootcas.org.uk)

OK

A flower is broken down into smaller parts.

Common Flower Parts



OK

* 23. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in decomposition?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

* 24. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching decomposition?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

25. In a sentence or two describe a lesson you taught that includes **decomposition** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

Concept: Patterns

Patterns are described as "spotting similarities and common differences. By identifying patterns we can make predictions, create rules, and solve more general problems" (Barefoot Computing, n.d.). When children learn to recognize repeating melodies in music or phrases in stories, when they see similarities and differences in data collected in science, or rules of number sequences in math, they are identifying patterns (www.barefootcas.org.uk).

OK

Students look for patterns to construct a Sierpinski Triangle.



OK

* 26. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in recognizing patterns?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

* 27. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching patterns?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

28. In a sentence or two describe a lesson you taught that includes **patterns** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

Concept: Abstraction

"Abstraction is about simplifying things – identifying what's important without worrying too much about detail" (Barefoot Computing, n.d.) For example, math word problems involve students identifying the key information and learning how to present the problem in the language of numbers. In geography, students use maps to represent an area without showing the complexity of the environment. Other examples of abstraction include creating a story plan or mind map where details are left out, making notes and charts of the most important properties in science, creating a presentation with key points on a topic, or making an argument with supporting information (www.barefootcas.org.uk).

OK

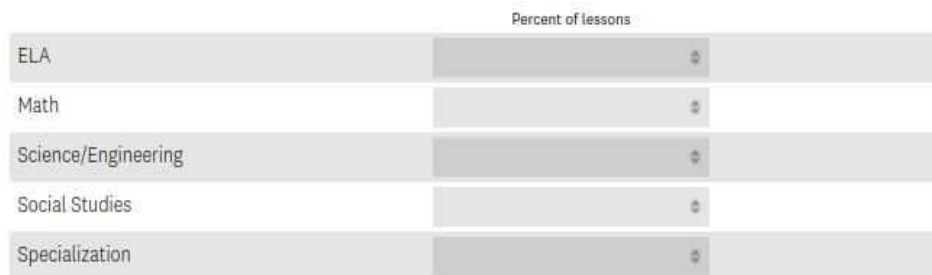
A timetable is an abstraction of the school day, much is omitted to summarize this occurrence.

	1	2	3	4	5	6
MONDAY	Science	English	Math	Lunch	History	Gym
TUESDAY	Art	Science	English	Math	Lunch	History
WEDNESDAY	History	Gym	Science	English	Math	Lunch
THURSDAY	Math	Lunch	History	Art	Science	English
FRIDAY	English	Math	Lunch	History	Gym	Science

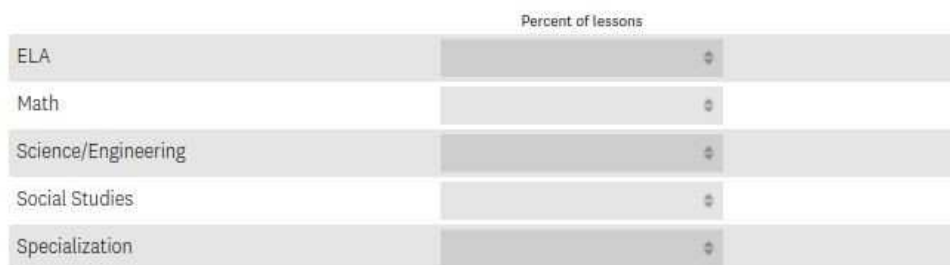
(iStock.com/mariaflaya, 2019)

OK

* 29. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in abstraction?



* 30. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching abstraction?



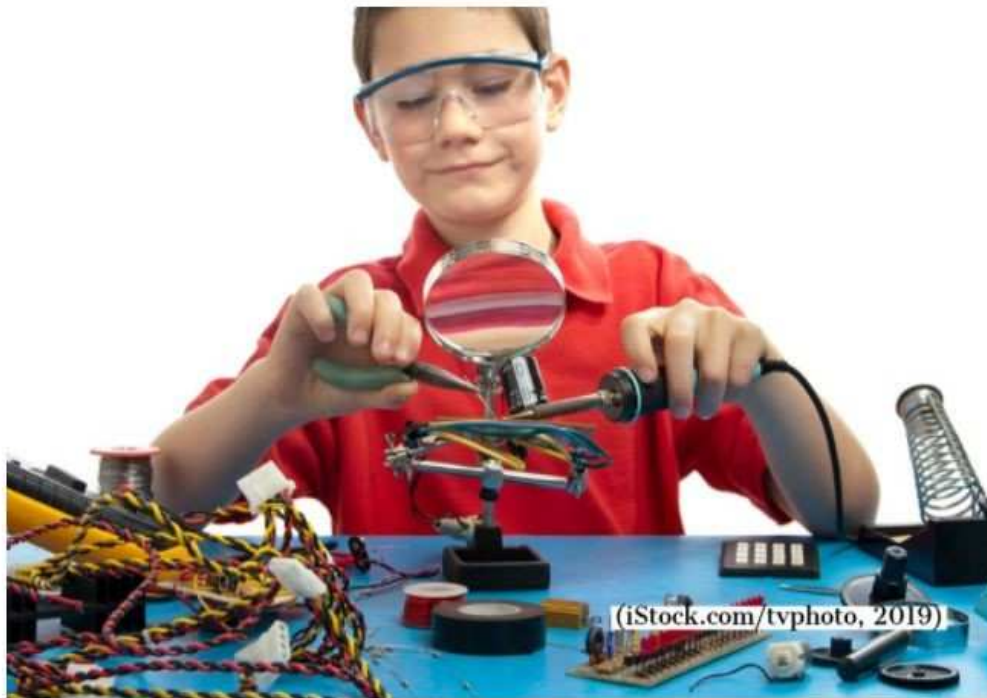
31. In a sentence or two describe a lesson you taught that includes **abstraction** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

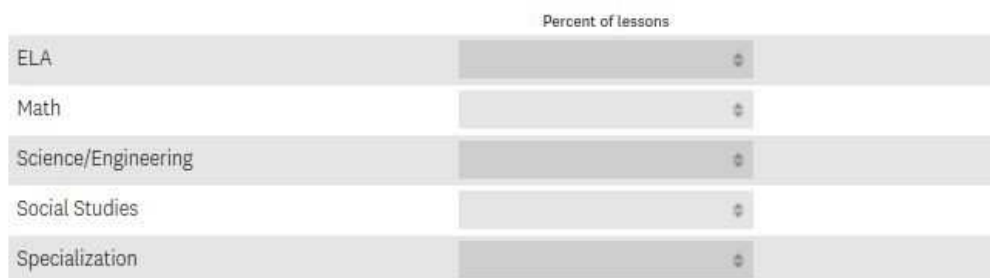
Approach: Tinkering

Tinkering is an approach to thinking where you try things out. This is the play based, exploration and an experimentation phase of learning. Examples include children trying things out through role playing, exploring, asking why and how questions, figuring out how things work, building and creating, and testing new ideas (barefootcas.org.uk).

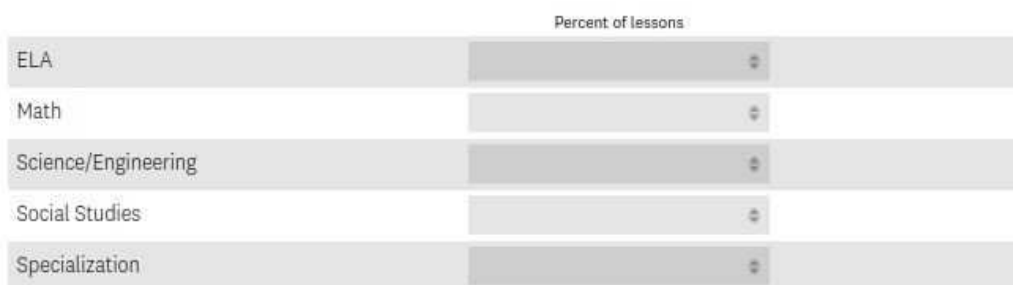
OK



- * 32. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in tinkering?



- * 33. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching tinkering?



34. In a sentence or two describe a lesson you taught that includes **tinkering** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

Approach: Creating

"Creating is about planning and making things. Some endeavors involve various media each providing an outlet for creative expression. Software and digital media allow scope for creativity and, by mastering software tools and digital devices, we develop confidence, competence and independence which we can use playfully, experimentally and purposefully in the expression of our ideas and insights" (Barefoot Computing, n.d.). Examples include students making games, animations, quizzes, models, artwork, toys, and inventions (www.barefootcas.org.uk).

The girl is creating a camouflage shirt to blend into the background.



- * 35. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in creating?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

- * 36. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching creating?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

37. In a sentence or two describe a lesson you taught that includes **creating** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

Approach: Debugging

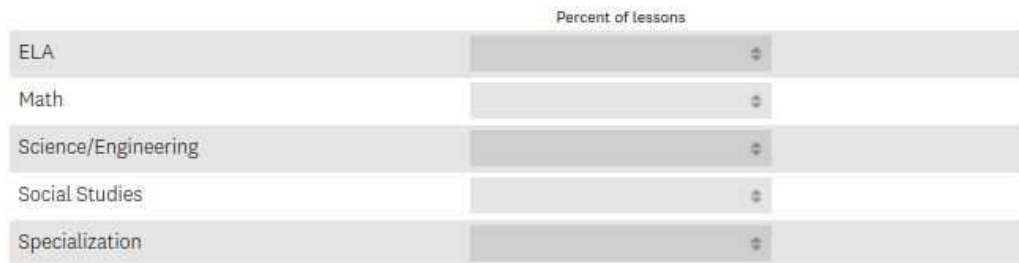
Debugging is an approach to thinking where students are finding and fixing errors through a process such as predicting what should happen, finding out exactly what did happen, working out where something went wrong, and fixing it. Examples include students find and fix errors in their work and their peers' work and find opportunities for improvement(www.barefootcas.org.uk).

The girl just debugged her problem with the robot by fixing the algorithm.

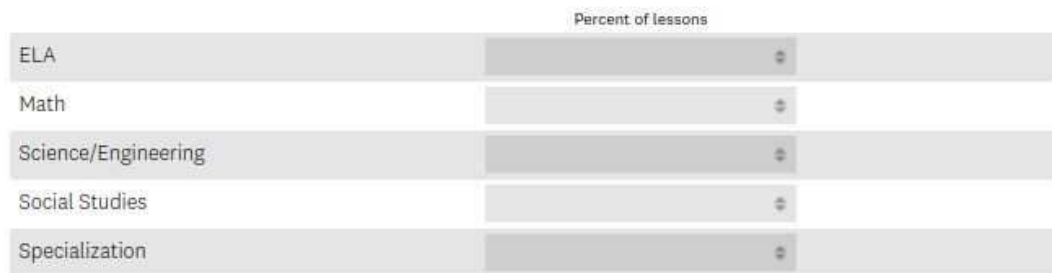


(iStock.com/Zinkevych, 2019)

* 38. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in debugging?



* 39. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching debugging?



40. In a sentence or two describe a lesson you taught that includes **debugging** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

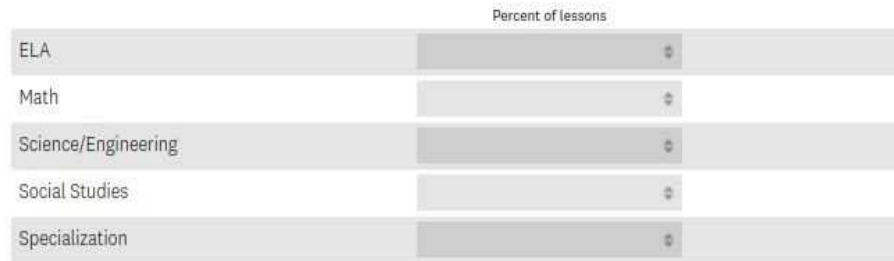
Approach: Perseverance

Persevering is an approach to thinking where you never give up, you are determined, resilient, and tenacious. Examples occur in music, sports, and dance where students need to practice, train, and rehearse to improve their skills. Solving puzzles, building complex models, participating in activities that take many days, tackling difficult problems while experiencing confusion are other examples where students persevere (www.barefootcas.org.uk).

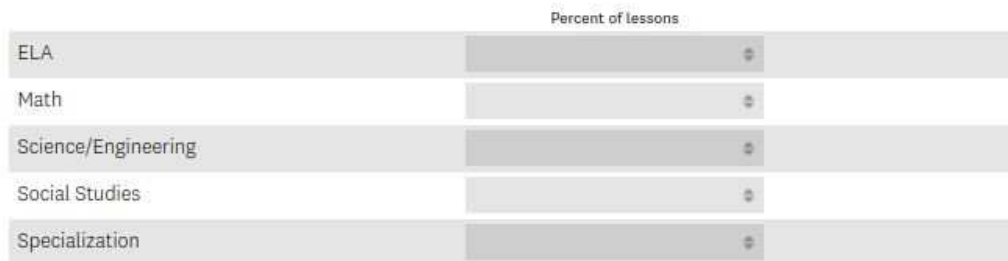
The students persevere with learning a new piece of music.



- * 41. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in perseverance?



- * 42. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching perseverance?



43. In a sentence or two describe a lesson you taught that includes **perseverance** in the teaching and learning process. (optional)

Computational Thinking in Elementary School Classrooms

Approach: Collaborating

Collaborating is an approach to thinking where you work with others to ensure the best result. Examples include students taking turns, working together, listening to each other, providing feedback, helping each other, and working as teams (www.barefootcas.org.uk)

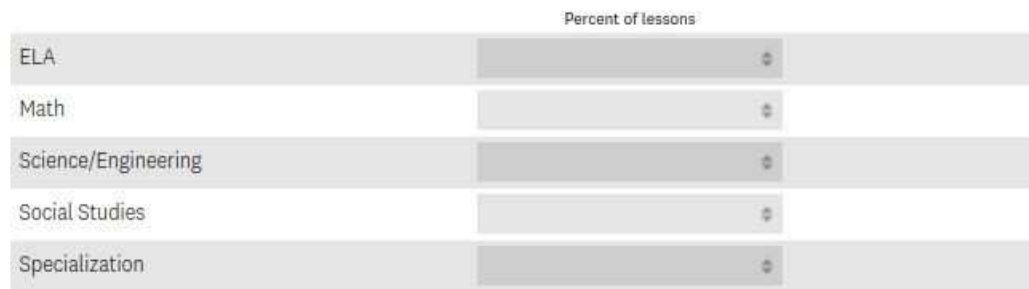
The students collaborate with their project.



* 44. Think about the last lesson you taught in each of the following subjects. What percent of that lesson did you spend teaching about or having the students engage in collaboration?

	Percent of lessons
ELA	<input type="text"/>
Math	<input type="text"/>
Science/Engineering	<input type="text"/>
Social Studies	<input type="text"/>
Specialization	<input type="text"/>

* 45. Think about each subject curriculum as a whole (year long). On average what percent of time in that subject do you spend teaching collaboration?



46. In a sentence or two describe a lesson you taught that includes **collaboration** in the teaching and learning process. (optional)

47. How often have you participated in professional development on computer science, computing, or computational thinking?

- ☐ I have not had any professional development.
- ☐ 1-3 hours
- ☐ 4-6 hours
- ☐ 7-10 hours
- ☐ 11-20 hours
- ☐ More than 20 hours

48. Could you describe the professional development you have received?

49. What is your interest level in attending professional development for computational thinking?

- ☐ Extremely interested
- ☐ Very interested
- ☐ Somewhat interested
- ☐ Not so interested
- ☐ Not at all interested

50. Do you think computational thinking should be a priority in education?

- ☐ Strongly agree
- ☐ Agree
- ☐ Neither agree nor disagree
- ☐ Disagree
- ☐ Strongly disagree

51. Does your school district encourage computational thinking?

- ☐ Yes
- ☐ No

Please comment on your choice.

52. What supports would you like to see in the future to help you integrate computational thinking into your curriculum? (optional)



53. Computational thinking is relevant to my current job functions.

- ☐ Strongly agree
- ☐ Agree
- ☐ Neither agree or disagree
- ☐ Disagree
- ☐ Strongly disagree

Computational Thinking in Elementary School Classrooms

Now that you know more about computational thinking...

54. Which statement best describes your concern about teaching computational thinking curriculum in elementary school?

- ☐ "I've heard something about it, but other responsibilities take priority"
- ☐ "This seems interesting, and I would like to know more about it."
- ☐ "I'm concerned about the changes I will need to make in my routines."
- ☐ "I'm concerned about how much time it takes to get ready to teach with this new approach."
- ☐ "How will this new approach impact my students?"
- ☐ "I'm looking forward to sharing some ideas about it with other teachers."
- ☐ "I incorporate computational thinking skills into my lessons now and have ideas about how to do it better."

55. How would you rate your confidence level in teaching computational thinking?

- ☐ Extremely confident
- ☐ Very confident
- ☐ Somewhat confident
- ☐ Not so confident
- ☐ Not at all confident

56. Overall, how often do you think you apply computational thinking in your classroom?

- ☐ Most lessons each day
- ☐ A few lessons each day
- ☐ One lesson a day
- ☐ A few lessons each week
- ☐ A few lessons each month
- ☐ A few lessons each year
- ☐ Never

Computational Thinking in Elementary School Classrooms

Thank you for learning and sharing with us! Please watch the following videos (<https://www.youtube.com/watch?v=sxUJKn6TJOI&feature=youtu.be> & <https://www.youtube.com/watch?v=VFcUgSYyRPg>) and reflect on your current practices in the classroom. Consider how you could more explicitly integrate computational thinking into your lessons. Click on the link to print PLU documentation.

DONE