Mutually Supportive Mathematics and Computational Thinking in a Fourth-Grade Classroom

Janet Fofang, David Weintrop, Margaret Walton, Andrew Elby, and Janet Walkoe
bihjane@umd.edu, weintrop@umd.edu, mwalton@umd.edu, elby@umd.edu, jwalkoe@umd.edu
University of Maryland

Abstract: Educators have argued for the value of integrating the big ideas of computing, captured under the term computational thinking (CT), into subjects and classrooms across the K-12 spectrum. Working within a researcher-practitioner partnership, we developed activities designed to integrate CT into a fourth-grade mathematics classroom using the Sphero robot. The goal was to create activities where the mathematics served as context to employ CT strategies and the CT helped deepen mathematics engagement. Here, we present two vignettes from this work. One shows what mutually-supportive CT and mathematics can look like in a fourth-grade classroom, and a second illustrates how both CT and mathematics can be present but not mutually supportive. This work advances our understanding of what it looks like to integrate CT into mathematics in resource- and time-constrained public-school elementary classrooms.

Introduction

With the growth of computing and computational thinking (CT) in STEM careers and in public debates of socio-scientific issues, it is critical that all children have strong preparation in computing and computational thinking (CT). However, many learners, especially those in under-resourced schools, have relatively little opportunity to engage in meaningful CT learning early in their schooling. One approach for addressing this issue is to integrate CT learning opportunities into existing domains. There are compelling practical and theoretical reasons for taking such an approach, including the potentially mutually-supportive nature of CT and disciplinary content with respect to learning. Further, integrating CT into existing coursework can address issues related to school resources and teacher expertise (Weintrop et al., 2016). As computing plays an increasingly important role both within and beyond the classroom, it is critical to provide opportunities for all learners to develop essential foundational CT skills and practices.

In this work, we investigate ways to integrate CT into fourth-grade mathematics classrooms. As part of a Researcher-Practitioner Partnership (Coburn, Penuel, & Geil, 2013) with an urban school district, we developed a mathematics curriculum in which learners use the Sphero robot as a way to explore mathematical ideas and employ CT concepts and practices. Key decisions in the design of the curriculum, including the technology (Sphero) and orienting mathematics curriculum (Eureka Math), were based on existing district resources. Importantly, the mathematics standards and accountability measures in place in the school were not affected by this study. In this way, the intervention was not an add-on enrichment program, nor did it replace the mathematical learning goals. Instead, it was designed to fit within school constraints. Further, by working with under-resourced public schools and by integrating CT into mathematics contexts, the approach can reach students who have had little access to computing education opportunities (Margolis, 2008; Ryoo et al., 2013).

Our goal is to investigate what happens when CT is introduced into a fourth-grade mathematics classroom. This paper presents two vignettes highlighting two very different roles CT ended up playing in our lessons. In the first vignette, CT and mathematics co-occur in mutually-supportive ways. In the second vignette, mathematics and CT are present but distinct, a fact recognized both by the researchers and the learners. In presenting these two vignettes side-by-side, we provide first an existence proof for mutually-supportive CT and mathematics with younger learners within the constraints of public schools, then an example of how integration can “drift” into disconnected co-existence. Collectively, this work provides new insight into the nature of integrating CT into elementary classrooms and advances our understanding of the challenges associated with doing so.

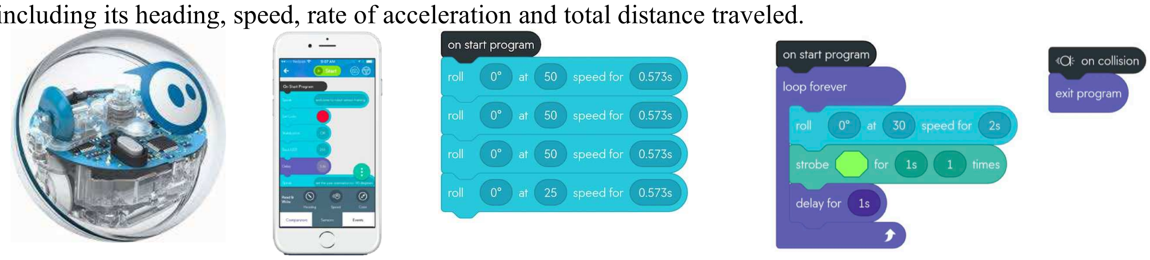
Prior work

Wing’s (2006) call to action around the CT reignited the conversation of the broad potential for computing in educational contexts. This idea has a long history in Constructionist literature, where researchers have spent the last half-century exploring CT in various forms (diSessa, 2000; Harel & Papert, 1991; Kafai & Resnick, 1996; Papert, 1980). In the decade since Wing’s reintroduction of the idea of CT, many have stepped forward to try to provide a concise and useful definition for CT (Grover & Pea, 2013; National Research Council, 2010; Shute et al., 2017), although consensus around where the bounds of CT lie has yet to emerge. In this work, we conceptualize CT as the set of concepts and practices associated with using computing to help solve problems, including problem decomposition, abstraction, pattern recognition, algorithm design, and skills and knowledge associated with programming (Dong et al., 2019).

The work presented here builds on the growing body of working advocating for an integrated approach to bringing CT into the classroom (Barr & Stephenson, 2011; Israel & Lash, 2019; Sengupta et al., 2013; Settle et al., 2012; Weintrop et al., 2016; Wilensky, Brady, & Horn, 2014). Central to this approach is the idea that CT and disciplinary content can be mutually-supportive, meaning that disciplinary content can provide a meaningful context to employ CT concepts and practices and that doing so can deepen disciplinary understanding. The idea of computing serving as a context for mathematical thinking is not new. In 1972, Papert wrote a paper titled "Teaching Children to be Mathematicians Versus Teaching About Mathematics" in which he explored the idea of kids programming as a way for them to engage in mathematical ideas and practices. This work produced a large body of research on how programming, computers, and later CT can serve as a productive context for exploring mathematical concepts (e.g. Abelson & diSessa, 1986; Kaput, Noss, & Hoyles, 2002; Noss & Hoyles, 1996; Roschelle, Kaput, & Stroup, 2000; Wilensky, 1995). With broadening ideas about what it means to think mathematically (e.g., Carpenter, Franke, & Levi, 2003) there is growing recognition of how mathematical thinking can be elicited through CT (Calao et al. , 2015; Israel & Lash, 2019; Pei, Weintrop, & Wilensky, 2018; Wilkerson-Jerde & Wilensky, 2015). Finally, this work draws on prior findings around the potential for robots and physical computing to serve as a way for younger learners to engage with CT (Bers et al., 2014; Chalmers, 2018; Rusk et al., 2008; Shumway et al., 2019).

Meet Sphero.Math

Working with a computer science specialist, a curriculum developer, and a mathematics teacher in our partner district, we created a 15-lesson fourth-grade mathematics curriculum called Sphero.Math. The activities in the curriculum challenge students to explore mathematical ideas by programming a Sphero robot (Figure 1a). The Sphero is a spherical robot that can be programmed via tablets using both graphical and textual programming languages. Students can programmatically control Sphero using a tablet or smartphone (Figure 1b). For the Sphero.Math curriculum, students programmed their Sphero using the block-based programming modality (Figure 1c and 1d). Block-based programming is a graphical programming approach that uses a programming-command-as-puzzle-piece metaphor to support novices in having early programming success (Bau et al., 2017; Weintrop & Wilensky, 2015). The Sphero programming environment includes blocks to control the Sphero's movement (e.g. roll, stop) and appearance (e.g. strobe lights), while also including conventional programming constructs such as looping structures, variables, and logic operators. The basic movement command for the Sphero is the `roll` command which takes three inputs: heading, speed and time, resulting in a command that reads: `roll 90° at 50 speed for 2s`. Sphero uses an event-based programming model, making it easy to define behaviors in response to events, such as detecting a collision. Further, the Sphero includes various sensors and can report out features including its heading, speed, rate of acceleration and total distance traveled.



(a) (b) (c) (d)

Figure 1. The Sphero robot (a), programming environment (b), and two sample programs (c) and (d).

The Sphero.Math curriculum is aligned to fourth-grade mathematics standards and focuses on concepts such as data, place-value, measurement, rounding, and algorithms for addition and subtraction, while also engaging learners in core CT practices like developing algorithms, creating abstractions, and debugging. Each lesson is aligned to Eureka Math, which is the fourth-grade math curriculum used by schools across the district. The Sphero.Math lessons usually involved programming the Sphero to move fixed distances or follow a particular pattern designed to elucidate a math concept. Students used programming concepts that could control robot movement by either holding the speed constant and varying the time or holding time constant and varying the speed to determine the distance.

Methods

Setting and participants

This study took place in one fourth-grade classroom in an urban elementary school in the Mid-Atlantic region of the United States. The school is relatively racially diverse (51% White, 17% Latinx, 15% Black, 9% Asian, 8% Mixed Race) with 19% of the students in the school designated as English Language Learners. We worked closely with the classroom teacher and the school technology coach to bring the curriculum into the school. The class had a total of 21 students and the teacher had students work in pairs for the Sphero.Math activities. Each pair was given an iPad, a Sphero and whatever other materials were needed for the day's lesson. The lessons we observed were taught in the final month of the school year. By that point, students were familiar with the structure of Sphero.Math activities and had experience working with the Sphero robots. For data collection, the teacher selected two pairs of students to serve as focal groups. These groups were asked to work outside of the classroom to decrease ambient noise and ensure non-consenting students were not captured on video.

Data and analytic approach

The primary data source for this study is the video that was collected during the Sphero.Math lessons from the two focal pairs. Each pair was video recorded using a stationary camera set up next to the groups' work area. Further, each focal student wore a head-mounted camera throughout the activities to provide a student-centric perspective of the activity. This resulted in three videos for each focal pair (two head-mounted videos and the stationary camera). Data was collected as students worked through Sphero.Math lessons during their normal hour-long math classes. Along with observing the video data, a researcher briefly interviewed the students at various points throughout the activities as a means of gaining further insight into the nature of the students' experiences with the curriculum. At the end of each lesson, short interviews were conducted asking students to reflect on the just-completed activity.

After each data collection session, two researchers on the project reviewed all of the collected video looking for portions of the activity that seemed particularly interesting and worthy of closer analysis. Both researchers have prior experiences as teachers and were actively involved in the creation of the curricular materials. In particular, the researchers were looking for the range of ways in which the "integration" of CT and mathematics played out. We chose one vignette to highlight the type of integration that most resembled what we envisioned and one vignette that was more typical. In our brief analyses, we focus on the mathematical content and practices and the CT concepts and practices present in the discourse (e.g., Brennan & Resnick, 2012; Grover & Pea, 2013; Weintrop et al., 2016).

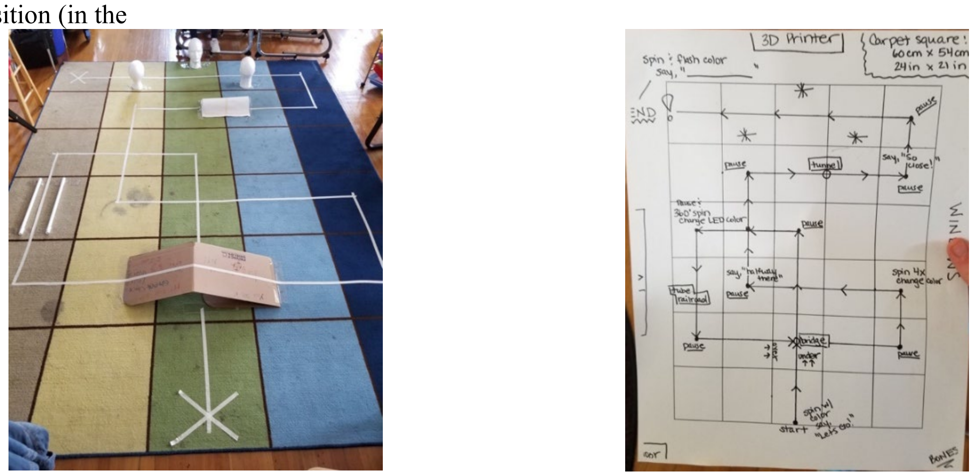
Findings

This section is divided into two subsections. First, we present an episode where we observed the two fourth-grade learners engaged in a Sphero.Math activity where the mathematics and CT were mutually-supportive. In particular, we show how the mathematic and CT co-occur and are mutually informing. In the second vignette, we document a second pair of learners working through a different Sphero.Math activity where both CT and mathematics are present but have little conceptual integration or interaction. Each section begins with an introduction to the activity, followed by a description of how the students carried out the task and a discussion of the CT and mathematics on display.

Class obstacle course: Mutually-supportive mathematics and computational thinking

The summative activity for the Sphero.Math curriculum was for students to program their robots to complete an obstacle course defined by a series of lines of masking tape laid out on the classroom carpet (Figure 2a). The course included numerous obstacles and challenges that required changing position and speed as well as making sounds and changing the Sphero's appearance. However, unlike conventional robotics maze challenges, during the development of the program, students were not able to run the robot on the course directly. Instead, the teacher provided a reference sheet showing the route and the dimensions of the carpet tiles upon which the obstacle course had been created (Figure 2b). In order to complete the course, students had to calculate the length of each course segment using the size of the carpet patch as the unit (i.e. a segment may be 1.5 carpet patches long). Once students completed their program, they could then run it on the actual obstacle course to see how far their robot could go. Along with the reference sheets, students were given yardsticks to help calibrate their programs.

Both mathematical and CT practices are on display in this lesson. The activity aligns with fourth-grade Common Core Mathematics Standards asking learners to "Solve problems involving measurement and conversion of measurements" (CCSS.Math.Content.4.MD.A.1-3). In this case, students are measuring the length of segments of the course and converting between conventional measures (inches and centimeters) and Sphero-defined distances (defined by speed and duration). At the same time, learners engage in numerous CT practices, including decomposition (in the form of breaking the course down into discrete sections), creating algorithms (in the form of defining the steps to be followed), and pattern recognition (in identifying which portions of the maze repeat and can thus be encapsulated in a function or nested within a repeating control structure).



(a) (b)

Figure 2. The maze to be completed (a) and the guide students were given to program their robot (b).

Students programming their Sphero to complete the first segment of the course

At the start of the activity, the two students in the focal pair recognize that they need to get their Sphero to travel 24 inches (the height of one square). The plan is to use that command as the building block for their program. This is significant in that they recognize that having a single command that moves 24 inches could be used as a form of unit of measure: 1 square. To figure out how to get their Sphero to move this fixed amount, they line up their Sphero at the beginning of the yardstick and start by coding the Sphero to move at speed 50 for .7 seconds. The resulting command reads: `roll 0° at 50 speed for .7s`. When they run their first program, the Sphero goes 32 inches, as measured by the yardstick. This prompts the second student to suggest shortening the duration of the roll command to .5. Instead, the first student programs the Sphero to travel for .3 seconds. When lined up against the yardstick, the Sphero only travels six inches in .3 seconds. As a result, Student One says, "*Oh yeah, .5*". At .5 seconds, the Sphero travels just under 24 inches, prompting Student One to say, "*I think [the teacher] would give us that.*" Student Two replies, "*No, because if it kept going like that…"* then trailing off as his partner says, "*Ok then .573.*" Student One changes the code to make the Sphero go for .573 seconds and they test the program again. The Sphero travels exactly 24 inches, prompting student Two to say, "*Right on the dot.*"

Having defined a program to traverse one square, the students go back to the map worksheet and count the number of squares needed to complete the first maze segment. The students determine that they need to go the length of 3.5 squares to complete the first part of the maze. After some brief discussion, the students realize that in order to travel one half of a square, they will need to "*cut the speed in half*". The result is a program that calls their previously defined square command 3 times, followed by a fourth call where the speed is cut in half. Their Sphero program at this point in the activity is shown in Figure 2c.

Covariation, proportional reasoning, iterative development, and Sphero programming

Throughout this activity, we can see students demonstrating an understanding of covariation and proportionality mediated by the task of programming the Sphero. Students employing the idea of covariation can first be seen as the pair is trying to get their Sphero to travel exactly 24 inches. When their first attempt travels too far, they decrease the duration. When it then does not roll far enough, they increase the duration. Both of these modifications demonstrate an understanding of the relationship between rate and time and how it impacts distance traveled. A second demonstration of this understanding can be seen at the end of the episode when the Sphero needs to travel half of a square, resulting in the suggestion to cut the speed in half. It is important to note in the first part of the vignette; the students are manipulating time to change the distance traveled and in the second they modify the speed. This suggests an understanding of proportionality and the relationship between speed and duration as it relates to distance.

At the same time this mathematical engagement was happening, there are a number of CT practices on display. First, the students' decision to write a program to travel 24 inches and then re-use it as they navigate the course is an example of decomposition and the creation of a modular solution. Second, we can see the learners employing an iterative development approach, where they proposed a potential solution and then revised their proposed solution based on previous runs of the program. Finally, this mathematical expression and enactment of CT practices happened while the learners were expressing their ideas in a form that a computer could execute. In other words, the students were programming, and thus performing another important aspect of CT. This is one way in which CT and mathematics were mutually supportive within this vignette. The Sphero programming provided a context for applying proportional reasoning, while the challenge of defining measurements and precision situated iterative development, problem decomposition, and program construction.

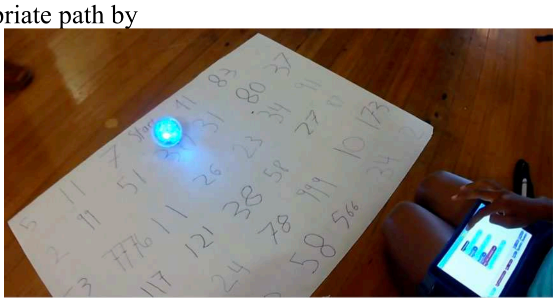
Precision, pattern recognition, and being off by just a little

A second example of the mutual supportiveness of mathematics and CT can be seen in how the learners in this vignette attend to issues of precision and the implications of being imprecise. When the students authored a program that went close to 24 inches but stopped a little short, one student's first reaction was *"I think* [the teacher] *would give us that".* This comment suggests the learner is deferring to the teacher as the arbiter of accuracy and that if the teacher thinks it is close enough, that is good enough. However, the second student responds, "*No, because if it kept going like that…*" before trailing off. We view this second comment as the fourth-grade learner attending to issues of imprecision and the implications of having small errors in commands that will be repeated. The goal of creating a program to run exactly 24 inches is so that they can use the single 24-inch-instruction as the primary building block for their program. As can be seen in Figure 2c, their solution to completing the first segment of the course is to repeat their 24-inch-command three-and-a-half times. In this second utterance, the student recognizes how being off by a little at each step will result in a much larger difference in the end. In this way, he is applying multiplicative logic, showing how the small error will grow over the course of the program.

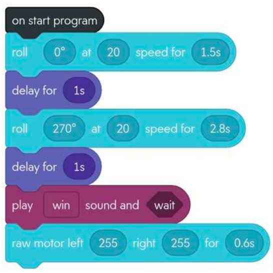
Noticing this potential issue of a small error being magnified as the logic is repeated also demonstrates a number of CT concepts, including pattern recognition, algorithm development, and knowledge of how iterative logic behaves. Further, in this example, we can see how the programming context gave concrete meaning to the measurement they were creating as it pertained to the task of completing the obstacle course. While the first student may have been right that the teacher would have "given it" to them as close enough, the obstacle course itself would not have.

Prime number path: Mathematics and computational thinking in Parallel

The second vignette we present in this work serves to demonstrate an instance where mathematics and CT are both present but play parallel roles within a single activity rather than being mutually supportive. This activity has learners explore the idea of prime and composite numbers. At the start of the activity, students were given a large sheet of paper and asked to create a grid and then write a number in each square. In filling out the squares, students were instructed to create a "prime number path", meaning the numbers should be arranged such that there is a contiguous path of prime numbers leading across the page. Students indicated a start and end point for their paths. A successful pattern on the paper meant there was one path with only prime numbers from a start to a finish point (Figure 3a). Once their prime number path was approved by the teacher, students were then challenged to program their Sphero to navigate the chart by only rolling across prime numbers and never "touching" a composite number (e.g. Figure 3b). Once they had a program to navigate their own prime number path, students swapped sheets of paper with their classmates to try to solve their peers' prime number paths. This social aspect encouraged groups to create more complex pathways and use larger, less common numbers. Unlike a typical maze-navigation activity, this exercise engaged students to think carefully about prime and composite numbers as they decided on the appropriate path by



(a)



(b)

Figure 3. A Sphero navigating a prime number path (a) and a prime number path program (b).

selecting prime numbers to complete the maze. During the design of this activity, students constantly referred to textbook definitions of prime and composite numbers. They used these definitions to produce large composite and prime numbers, which challenged their classmates' understanding of composites and primes and the strategies that can be used to determine if a given number is prime.

Prime or composite? Programming or mathematics?
Central to this activity was the task of students deciding if a number was prime or composite. This was the case both when students created their own prime number paths as well as when they tried to figure out the path to take through their peers' mazes. When students started on a new board, the first thing they did was try and find the path. To make the Sphero complete this task, the students had to place the Sphero on the start square, orient the Sphero so it was pointing in the right direction and then count out the number of squares the Sphero would travel before needing to change direction. As they worked through the maze, students would attend to distances and angles as they went. In many cases, students did a large amount of guessing and checking in order for their Spheros to make their way through the maze, incrementally working their way through the prime number path, a little bit at a time.

This activity included learners reasoning about both the focal mathematics content (characteristics of prime and composite numbers) as well as CT practices (e.g. programming, incrementally developing a program, debugging), but the activity was constructed in such a way that students were only engaged with one of these aspects at a time. Pairs either first figured out the prime number path and then programmed it or moved back and forth between finding the next prime number and then programming their Sphero to get to that square. In both cases, the CT and mathematics were not mutually supportive but rather two independent aspects of the same activity. This separation was evident to the students who went through this activity. When interviewed about the activity afterward, one of the focal students verbalized this lack of integration, saying *"well, they were kind of two different parts because there was more like finding the path and coding it".* Her partner echoed this sentiment, saying the mathematics was related to "*finding the prime numbers*" and not the Sphero programming component of the activity.

Discussion

Delivering on the potential of integrating CT across the curriculum
Much has been written about the potential and power of integrating CT across the curriculum (e.g. Barr & Stephenson, 2011; Sengupta et al., 2013; Weintrop et al., 2016). One goal of this work is to document the fact that not all CT integration is created equal. While there are countless ways to integrate technology, programming, systematic problem solving, or other aspects of CT into a curriculum, that does not necessarily mean the revised curriculum will result in a deeper engagement with the materials or better learning outcomes. This is not to say that such integrations are not potentially valuable. For example, students really enjoyed the prime number path activity, especially compared to alternative classroom activities for identifying prime and composite numbers. But we raise this point to highlight the need for more sophisticated measures of levels and effectiveness of integration and the need to develop a language around the nature of such integration and the role(s) that CT is playing both as a learning object on its own and in support of disciplinary learning. As the CT literature continues to mature and the number of projects presenting this type of mutually-supportive work expands, developing this critical lens and reliable language around which to discuss and evaluate such efforts is essential to best deliver on the promise of the integrated CT approach.

The role of the teacher
As we strive towards creating mutually supportive integrated CT activities, the role of the teacher cannot be overstated. The above vignettes focus explicitly on student interactions with the curricular materials and supporting technologies. While this approach made some sense in the context of this specific classroom where this research took place due to the presence of the researchers, this is not usually the case. One essential role for the teacher is in helping facilitate and draw out the mutual supportiveness within CT integrated lessons. This is by no means an easy thing to do as a growing body of research is documenting the challenges associated with this task (Cabrera, 2019; Yadav, Stephenson, & Hong, 2017) as well as emerging strategies for helping support teachers in this role (Hestness et al., 2018; Ketelhut et al., 2019). Helping prepare teachers not just to bring the ideas of CT into the classroom but to do so in a mutually supportive way remains an important challenge for the field.

The importance of working within the current educational system

One of the central goals of this work was to explore ways of integrating CT into classrooms that fit within the current educational landscape. This means working with in-service teachers, in typical schools, and within the constraints imposed by the school administrators and district. In this case, that meant fitting with the allotted mathematics time and doing so in a way that aligned with state standards and district curricula. In making this commitment to working within the system, we seek to address one dimension of the issue of equity as it relates to computing: access. By aligning the CT materials with the mathematics instruction already taking place in classrooms across the district, we lower the barrier to adoption and provide an easier path into the classroom. Further, by conducting this work through a researcher-practitioner partnership, we had insight into the priorities and resources of the district, which shaped key decisions for this work, including the choice of grade, technology, and orienting curriculum. Collectively, these efforts result in a series of activities specifically designed to easily slide into existing classrooms within our partner district and ultimately introducing CT concepts and computing activities to fourth-grade learners who otherwise would not have had the opportunity at such an early stage of their K-12 education.

Conclusion

Computational thinking, and its associated concepts and practices, have widespread applicability both within classrooms and beyond. As part of the push to introduce these important skills to all learners, the approach of integrating CT into existing classrooms is becoming increasingly widespread. In this paper, we present two vignettes from our own work showing two very different roles that CT can play within a mathematics curriculum. In the first vignette, CT and mathematics content co-occur with the two being enacted in mutually-supportive ways. In such an activity, it is not possible to have students engage with only one side of the dual learning objectives. In the second vignette, both CT and mathematics are present but the two live as discrete parts of the same task. Aside from being a missed opportunity, such activities run the risk of being modified to remove one half of the material. The goal of this work is to provide an example of each of these forms of integration and begin a longer conversation around the various forms that CT integration can take. Further, in working in elementary grades through a researcher-practitioner partnership, this work serves as an empirical example of what integrated CT can look like in today's classrooms. This work reflects the start of what we hope will become a larger program of research exploring important and timely questions around the nature of integrated CT and ways to bring it into classrooms in effective and equitable ways.

References

Abelson, H., & diSessa, A. A. (1986). *Turtle geometry: The computer as a medium for exploring mathematics*. The MIT Press.

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.

Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: Blocks and beyond. *Communications of the ACM*, *60*(6), 72–80. https://doi.org/10.1145/3015455

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.

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. Presented at the American Education Researcher Assoc., Vancouver, Canada.

Cabrera, L. (2019). Teacher Preconceptions of Computational Thinking: A Systematic Literature Review. *Journal of Technology and Teacher Education*. 27(3), 305-333.

Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing mathematical thinking with scratch. In *Design for teaching and learning in a networked world* (pp. 17–27). Springer.

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., & Geil, K. E. (2013). *Research-Practice Partnerships: A Strategy for Leveraging Research for Educational Improvement in School Districts*.

diSessa, A. A. (2000). *Changing minds: Computers, learning, and literacy*. Cambridge, MA: MIT Press.

Dong, Y., Catete, V., Jocius, R., Lytle, N., Barnes, T., Albert, J., … Andrews, A. (2019). PRADA: A Practical Model for Integrating Computational Thinking in K-12 Education. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education - SIGCSE '19*, 906–912.

Grover, S., & Pea, R. (2013). Computational Thinking in K-12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38–43.

Harel, I., & Papert, S. (Eds.). (1991). *Constructionism*. Norwood N.J.: Ablex Publishing.

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.

Israel, M., & Lash, T. (2019). From classroom lessons to exploratory learning progressions: Mathematics + computational thinking. *Interactive Learning Environments*, *0*(0), 1–21.

Kafai, Y. B., & Resnick, M. (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world*.

Kaput, J., Noss, R., & Hoyles, C. (2002). Developing new notations for a learnable mathematics in the computational era. *Handbook of International Research in Mathematics Education*, 51–75.

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*.

Margolis, J. (2008). *Stuck in the shallow end: Education, race, and computing*. The MIT Press.

National Research Council. (2010). *Report of a Workshop on The Scope and Nature of Computational Thinking*. Washington, D.C.: The National Academies Press.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: Learning cultures and computers*. Dordrecht: Kluwer.

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

Pei, C. (Yu), Weintrop, D., & Wilensky, U. (2018). Cultivating Computational Thinking Practices and Mathematical Habits of Mind in Lattice Land. *Mathematical Thinking and Learning*, *20*(1), 75–89.

Roschelle, J., Kaput, J., & Stroup, W. (2000). SimCalc: Accelerating student engagement with the mathematics of change. *Learning the Sciences of the 21st Century: Research, Design, and Implementing Advanced Technology Learning Environments*, 47–75.

Rusk, N., Resnick, M., Berg, R., & Pezalla-Granlund, M. (2008). New Pathways into Robotics: Strategies for Broadening Participation. *Journal of Science Education and Technology*, *17*(1), 59–69.

Ryoo, J. J., Margolis, J., Lee, C. H., Sandoval, C. D., & Goode, J. (2013). Democratizing computer science knowledge: Transforming the face of computer science through public high school education. *Learning, Media and Technology*, *38*(2), 161–181.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, *18*(2), 351–380.

Settle, A., Franke, B., Hansen, R., Spaltro, F., Jurisson, C., Rennert-May, C., & Wildeman, B. (2012). Infusing computational thinking into the middle- and high-school curriculum. *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education*, 22–27.

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

Shumway, J. F., Clarke-Midura, J., Lee, V. R., Hamilton, M. M., & Baczuk, C. (2019). Coding toys in kindergarten. *Teaching Children Mathematics*, *25*(5), 316–317.

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.

Weintrop, D., & Wilensky, U. (2015). To Block or Not to Block, That is the Question: Students' Perceptions of Blocks-based Programming. Proceedings of the 14th International Conference on Interaction Design and Children, 199–208.

Wilensky, U. (1995). Paradox, programming, and learning probability: A case study in a connected mathematics framework. *The Journal of Mathematical Behavior*, *14*(2), 253–280.

Wilensky, U., Brady, C. E., & Horn, M. S. (2014). Fostering Computational Literacy in Science Classrooms. *Commun. ACM*, *57*(8), 24–28. https://doi.org/10.1145/2633031

Wilkerson-Jerde, M. H., & Wilensky, U. (2015). Patterns, Probabilities, and People: Making Sense of Quantitative Change in Complex Systems. *Journal of the Learning Sciences*, *24*(2), 204–251.

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM*, *60*(4), 55–62. https://doi.org/10.1145/2994591

Acknowledgments

This work is supported by the Spencer Foundation (#201900099). However, any opinions, findings, conclusions, and/or recommendations are those of the investigators and do not necessarily reflect the views of the Foundation.