Elementary Students' Use of Computational Thinking Practices Introduced Via Mathematics and Science

Kathryn M. Rich, Aman Yadav, & Emily Bouck
Michigan State University College of Education

Purpose

One barrier to broadening participation in computer science (CS) is that CS education is often relegated to the high-school level, and as an elective taken mostly by high-achieving students or students with previous computing experience (Warschauer & Matuchniak, 2010). One manner of addressing this problem is to include CS as part of the core curriculum earlier, thereby giving more students experience with CS and potentially inspiring confidence to take elective courses later. At the elementary level, including CS education in the curriculum brings particular challenges. Elementary teachers, while often enthusiastic about the idea of bringing CS to their students, have expressed a need to integrate CS into other subjects in order to fit it into their school day (Israel, Pearson, Tapia, Wherfel, & Reese, 2015; Rich, Yadav, & Schwarz, 2019).

Computational thinking (CT), defined broadly as the thinking processes used by computer scientists (Wing, 2006), provides an avenue for integrating CS ideas into other content areas, particularly STEM subjects. The Next Generation Science Standards (NGSS, 2013) include "using mathematical and computational thinking" as a practice, and a recent analysis of the Common Core State Standards for Mathematics in grades K-5 suggested substantial opportunity for integration of CT into elementary mathematics (Rich, Spaepen, Strickland, & Moran, 2019). The integration of CT into core subjects provides the additional advantage of exposing all students to CS, rather than only students to have access to or choose to take an elective (Weintrop et al., 2016; Yadav, Stephenson, & Hong, 2017).

Although integrating CT into core subjects in elementary school offers the potential for bringing early CS instruction to more students, there remain questions about if and how students who have been exposed to CT ideas integrated into core subjects will bring the ideas to bear in computational problems without a math or science connection. In this study, we ask: *In what ways do fifth-grade students who have been introduced to unplugged CT in the context of their mathematics and science instruction apply CT practices to a computational task unconnected to math or science?* We aim to generate preliminary evidence about the feasibility of beginning CS instruction in elementary school through integration of CT into core subjects.

Method

Theoretical Framing

We take an interpretivist perspective and seek to understand the ways in which students conceptualize how they use CT practices in their own problem solving. We used a task-based interview (Maher & Sigley, 2014) to engage students in a task designed to elicit computational thinking. The interview prompts used after students completed each part of the task were designed to elicit the participants' thinking as well as their own interpretations of how they used CT to complete the task.

Participants

The participants were 10 fifth-grade students from one classroom in an urban elementary school. The school population was 66% non-White and 59% of students received free or reduced lunch. The classroom teacher was a participant in an NSF-funded project designed to support elementary teachers in incorporating CT into their mathematics and science teaching. Through the teacher's participation in this project, the participants had been exposed to four CT practices throughout the school year: Abstraction, Decomposition, Debugging, and Patterns. The definitions, examples, and facilitating questions shown in Table 1 were displayed as posters in the classroom, and the teacher incorporated the ideas into his mathematics and science teaching in multiple ways, including prompting students to use one or more of the practices as they started a task and identifying and describing examples of when he saw students using the practices.

Table 1: Content of Classroom Computational Thinking Posters

Term	**Definition**	**Example**	**Facilitating Questions**
Abstraction	Focusing on the information I need while ignoring unnecessary details	Satellite view vs map view: What information do I need to get directions?	How can I simplify this problem? What is the important information? What information can I ignore? How can I only show the important information?
Decomposition	Breaking something down into smaller, more manageable parts	How do you eat a whole elephant? One bite at a time.	What is the big problem I am trying to solve? What are the different ways I could break this down? What parts are easy? What parts are difficult? What steps can I take?
Debugging	Finding and fixing errors or mistakes	Can you find the the mistake? 1 2 3 4 5 6 7 8 9	How can I tell whether or not my plan, model, or solution worked? How can I change something? Did the result match what I expected? Why didn't this work?
Patterns	Looking for similarities and patterns between things	What number is missing from the middle triangle? (Four triangles show a sum and two addends. One has a missing sum.)	What similarities or patterns do I notice? How can I describe the patterns? How can I use the pattern to make predictions or draw conclusions?

The ten participants were selected due to their parents returning a consent form and their presence in school on the day the interviews were conducted.

Task

We used an adaptation of the *Programming Lamps* task from the Bebras International Challenge on Informatics and Computational Thinking (Bebras, 2019). Students were presented with two-color chips in a 5-by-5 grid and told to imagine this was a grid of lamps, where the red chips were off and the yellow chips were on. Interviewers explained the grid could receive commands that changed the state of all the lamps in a row or in a column. For example, the command C2 would turn all the lamps in column 2 off if they were on and on if they were off. Figure 1 shows an example of the state of the grid before and after sending the command R3.

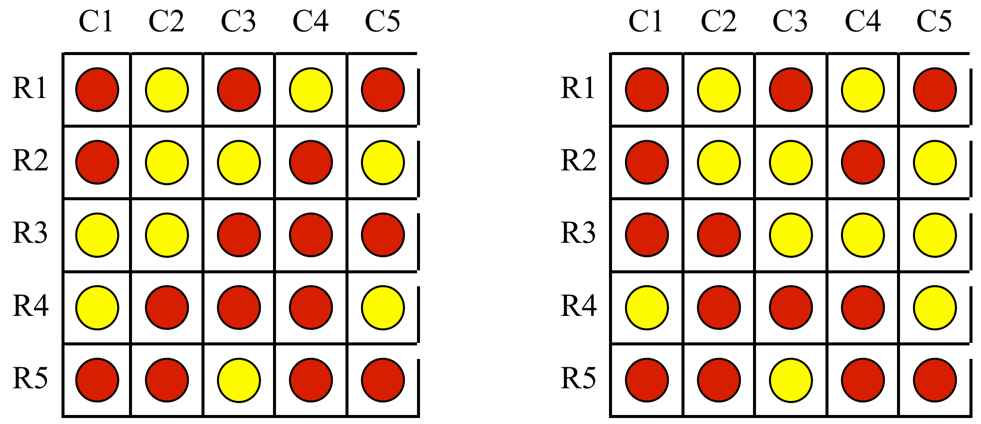
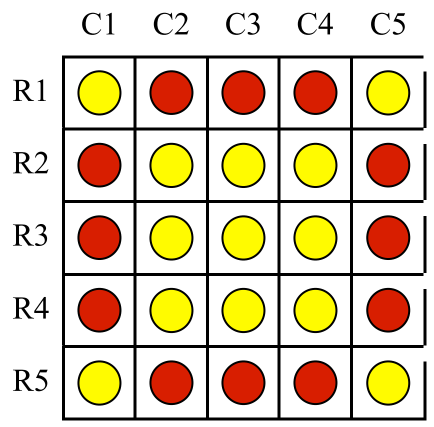


Figure 1: An example of the grid before (left) and after (right) sending the command R3.

In Part 1 of the task, students were asked to show, by flipping counters, what would happen when sets of five or six commands were sent to the grid. In Part 2 of the task, students were asked to turn all the lamps off and then determine which two of four sets of commands (given in Figure 2) could be sent to the grid so that it would look like Figure 2.



1. C1 C5 R2 R3 R4
2. R1 R5 C2 R3 R4
3. R1 R5 R3 R4 R5 C1 C5
4. C1 C2 C3 C4 C5 C1 C5 R1 R5

Figure 2: The goal appearance of the grid in Part 2 of the task (left) and the four sets of commands given to participants (right).

Procedure

The first two authors conducted the interviews one-on-one with students in the media center of their school. After gaining verbal assent from each child, the interviewer began the recording and explained the context of the task. The interviewer then asked the participant to show, using the counters, what would happen when a particular command was sent to the grid. The interviewer gave feedback and guidance on the participants' responses and continued asking them to illustrate what would happen with a single command until they could correctly execute a command independently. Then the participant was asked to complete Part 1. After completing Part 1, the interviewer asked the participant whether he or she used any of the four CT practices used in class this year. If the participant could not recall the CT practices, the interviewer listed them with brief definitions. Then the participant was asked to complete Part 2, and prompted again to describe how they used CT practices. Each interview lasted approximately 15 minutes.

Data Analysis

We recorded a brief description of each instance a participant identified themselves as using a CT practice. We then counted the number of students who said they used each practice. We grouped the descriptions of how students said they used the practices to gain a sense of the similarities or variations across students. We drew on other data sources from the broader project (classroom video and teacher interviews) to contextualize students' self-described use of CT.

Results

As shown in Table 2, all four practices were mentioned by at least one student, but there was clear variation across the practices in terms of the number of students who used them: ten used Patterns, nine used Debugging, five used Decomposition, and only one used Abstraction.

Table 2: Student Demographics and CT Practices Used

Student	Gender	Race	Abstraction	Decomposition	Debugging	Patterns
S1	F	White	x	x	x	x
S2	F	Black		x	x	x
S3	F	Black		x	x	x
S4	F	Black		x	x	x
S5	M	Black		x		x
S6	M	Black			x	x
S7	F	Black			x	x
S8	F	Black			x	x
S9	M	White			x	x
S10	F	White			x	x

Table 3 summarizes the ways in which students said they used each CT practice.

Table 3: Students' Descriptions of CT Practices

CT Practice	Description of Use	Number of Students
Abstraction	Ignoring the four sets of commands in Part 2 of the task because they were not important	1
Decomposition	Focusing on one row, column, or command at a time	5
Debugging	Noting that they fixed an error (without specifying the error)	4
	Noting they fixed a specific error, such as forgetting to flip over a chip	3
	Turning a lamp from on to off, or from red to yellow	2
	Breaking down the lists of commands and decoding each command	2
	Noting the task was hard to debug because she could not track what had been flipped	1
	Ruling out command sets in Part 2 of the task	1
	Noting that debugging is when something isn't finished	1
	Describing debugging as finding an answer	1
Patterns	Always turning red to yellow and yellow to red	5
	Seeing a similarity across two or more sets of commands in Part 2 of the task	2
	Evaluating success in Part 2 by comparing the pattern of counters on the grid to the goal pattern	2
	Noting that each row or column should be completely opposite after a command	1
	Noting that doing the same command twice would flip the counter twice	1
	Noticing that sometimes a row or column was completely on or completely off	1

The student who said she used Abstraction took an unusual approach to Part 2 of the task. Rather than attempt to determine which of the four given command sets would produce the desired pattern, she developed her own command set that produced the desired result. When asked whether she used Abstraction, she said she ignored the command sets because she thought figuring out her own would be easier than trying out the four given options. All five students who said they used Decomposition noted that they completed the task by focusing on one row, column, or command at a time.

Students' descriptions of how they used Debugging were varied. It was most common for students to say they used Debugging when they fixed something. Three students articulated a specific error they fixed, while four did not. One student commented that it was difficult to use Debugging because she could not keep track of what she had flipped over. Another said she used Debugging when she eliminated command sets from her choices in Part 2 of the task that she did not believe would produce the correct pattern on the grid. These comments all suggest some understanding of Debugging as a practice of fixing errors. Other student descriptions of how they used Debugging, such as saying they used Debugging when they broke down the problem or by finding an answer, do not seem to relate directly to a process of fixing errors.

The most common way students said they used Patterns was by consistently changing red to yellow or yellow to red. Three other comments about Patterns also had to do with consistency in commands or repetition of commands. Two students seemed to use a process of pattern-matching among possible solutions, as they said they used Patterns when they saw similarities across the sets of commands in Part 2 of the task. Two students said they used Patterns when they compared their grid of counters to the picture showing the goal state of the grid to determine whether they had found one of the correct command sets.

Discussion

The quantity and nature of students' descriptions of how they used the four CT practices are reflective of both how the practices were used in their classroom and the nature of the task itself. In our 90-minute lesson video of this classroom, the teacher explicitly points out examples of Decomposition, Debugging, and Patterns, but does not mention Abstraction. Moreover, the teacher noted in an end-of-year interview that he found Abstraction to be the most difficult CT practice to incorporate into his teaching. Additionally, the example given on the classroom Abstraction poster is eliminating trees, buildings, and other information irrelevant to navigation from a map, and the interview task does not provide much information that is irrelevant or easy to ignore.

Five students' consistent descriptions of using Decomposition by working command by command reflects a connection between Decomposition and working step by step through a problem. This is broadly similar to the classroom Decomposition poster's example of eating an elephant one bite at a time. It further suggests that when students encounter computer code, they may extend their understanding of Decomposition to writing or reading code one command at a time.

Seven of the ten participants were able to describe Debugging as a process of finding and fixing errors, consistent with the description on the classroom poster. The fact that only three of these students identified a specific error they fixed could reflect the challenge of Debugging their work on the task due to difficulty in tracking what they had already flipped (as pointed out by one of the participants). Two students' descriptions of Debugging as breaking something down may additionally reflect confusion between Debugging and Decomposition.

Finally, students' descriptions of how they used Patterns reflect a mix of looking across problems and contexts for similarities, as described on the classroom poster, and a more common understanding of a pattern as something that repeats.

Significance

This study contributes to the emerging body of knowledge about how CS education opportunities can be extended to broader populations. Elementary students from a racially and socioeconomically diverse school were introduced to four CT practices in the context of core subjects. Ten of these students were able to subsequently describe how they applied at least one of these practices to an unplugged computational task without a math or science context. Moreover, students' self-described use of the CT practices corresponded to some of the features of their classroom instruction, suggesting a link between their introduction to the practices in class and their use of the practices on the task. While some of the student descriptions reflected confusion about the practices, many, such as description of decomposing code into individual commands, debugging specific errors, and pattern-matching across possible solutions, were similar to how the practices might be used in coding. Future research is needed to examine how students further apply their knowledge of CT practices to coding and other plugged CS tasks, and whether and how their knowledge of CT practices affects their enrollment in future CS courses.

References

Bebras. (2019). "Programming Lamps." Task retrieved from https://www.bebras.org/?q=examples

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 and Education*, *82*, 263–279.

Maher, C. A., & Sigley, R. (2014). Task-based interviews in mathematics education. *Encyclopedia of mathematics education* (579-582). Springer Netherlands.

NGSS. (2013). “Next Generation Science Standards.” Retrieved from https://www.nextgenscience.org/search-standards

Rich, K. M., Spaepen, E., Strickland, C., & Moran, C. (2019). Synergies and differences in mathematical and computational thinking: Implications for integrated instruction. *Interactive Learning Environments*. Available online at https://doi.org/10.1080/10494820.2019.1612445

Rich, K. M., Yadav, A., & Schwarz, C. V. (2019). Computational thinking, mathematics, and science: Elementary teachers’ perspectives on integration. *Journal of Technology and Teacher Education*, *27*(2), 165–205.

Warschauer, M., & Matuchniak, T. (2010). New technology and digital worlds: Analyzing evidence of equity in access, use, and outcomes. *Review of Research in Education*, *34*(1), 179–225.

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.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35.

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