Action Fractions: The Design and Pilot of an Integrated Math+CS Elementary Curriculum Based on Learning Trajectories

Carla Strickland, Kathryn M. Rich, Donna Eatinger, Todd Lash*, Andy Isaacs,
Maya Israel**, and Diana Franklin
University of Chicago, Chicago, IL, USA
*University of Illinois at Urbana-Champaign, IL, USA
**University of Florida, Gainesville, FL, USA
{castrickland,kmrich,dmeatinger,aisaacs,dmfranklin}@uchicago.edu
toddlash.edu@gmail.com,misrael@coe.ufl.edu

ABSTRACT

The computer science (CS) education field is exploring several instructional strategies for teaching CS to children in elementary school. Strong arguments have been made for integration— constructing activities that not only teach CS, but use the CS to support learning in a core subject. Integrating CS materials into a specific curriculum is a non-trivial task that may unfairly burden elementary teachers, who are often generalists. Successful development and classroom implementation of integrated materials relies on many decisions about what, when, and how much subject matter to cover in relation to the main curriculum.

In this paper, we describe the design of a collection of 3rd and 4th grade (8–10 years old) integrated math+CS lessons. Our instructional materials use Scratch as a programming language and employ a learning trajectory approach to integrate the CS concepts of sequence, decomposition, repetition, conditionals, variables, and debugging into fractions content in a popular elementary mathematics curriculum. The integrated lessons are inserted throughout the main mathematics curriculum, providing multiple, non-continuous exposures to CS content. In addition, we present preliminary data from selected activities, including teacher feedback about the structure and impact of the math+CS instructional materials on their students' work.

CCS CONCEPTS

• **Social and professional topics** → **K-12 education**; **Model curricula**;

KEYWORDS

Curriculum Design, Integration, Scratch, K-12 Education

ACM Reference Format:
Carla Strickland, Kathryn M. Rich, Donna Eatinger, Todd Lash, Andy Isaacs, and Maya Israel, and Diana Franklin. 2020. Action Fractions: The Design and Pilot of an Integrated Math+CS Elementary Curriculum Based on Learning Trajectories. In *The 52nd ACM Technical Symposium on Computer Science Education (SIGCSE'21), March 13–20, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3408877.3432483

SIGCSE '21, March 13–20, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8062-1/21/03…$15.00
https://doi.org/10.1145/3408877.3432483

1 INTRODUCTION

As movements, such as CS4All, increasingly advocate for more CS instruction in K-12 schools, teachers and districts are seeking instructional materials to support CS implementation. While many instructional materials are available, including CS Unplugged [37] and code.org [40], most of these materials present CS as a standalone subject, particularly at the elementary level. There is growing momentum to introduce CS in an integrated manner, especially in the early grades (e.g., [14]; [33]). The rationale for this momentum includes several points: (1) research suggests learning CS can provide opportunities for students to engage in critical thinking (e.g., [10]) in a fun and motivating manner (e.g., [29]); (2) there is often natural overlap between CS and content areas such as mathematics (e.g., [22]), which makes for instruction that is mutually enhancing to both disciplines; (3) CS educational experiences have historically only been available to a subset of students who have access to enrichment activities, but by including CS in content areas, all students will engage in CS instruction ([38]); and finally (4) there are practical reasons for CS integration including lack of time for stand-alone CS courses in already packed school days (e.g., [12]).

The growing impetus to approach CS in an integrated manner, combined with the scarcity of instructional materials for integrated instruction, has placed the burden of instructional development on teachers [15]. However, development of instructional materials, and of integrated materials in particular, is a complex and difficult process. As the field continues to develop and iterate on instructional materials to suit the needs of a variety of teachers, students, and instructional contexts, detailed descriptions of instructional design principles, the needs those design principles fulfill, and how the principles are enacted in specific instructional materials will be beneficial. Such discussions will help to illustrate challenges in instructional materials development and establish the need to provide greater instructional support to teachers attempting integrated instruction.

In this paper, we present Action Fractions, a set of integrated fractions + CS activities for third and fourth grade students, along with the design principles underlying them. The materials utilize Scratch [19], a visual block-based language and programming environment designed for students in grade 3 (age 9) and older. We

begin by presenting prior work in Section 2. The design principles and the Action Fractions activities are presented in Sections 3 and 4. In order to show Action Fractions in the classroom, we then present methods of data collection with preliminary results (Section 5).

2 PRIOR WORK

To integrate CS into mathematics instruction, we considered two bodies of work. First, we drew upon elementary CS-only and integrated learning for pedagogical approach. Second, work on learning trajectories (LTs) in math and CS was used to design a cohesive sequence of activities that gradually built skills over two years.

2.1 Computer Science in Elementary School

CS instruction in elementary school has two distinct historical waves of instruction - the 1980-90's and post 2000. The modern era saw the rise of visual block-based languages (VBBLs) such as Scratch [11] and Alice [8]. With VBBLs, instruction began in informal settings through constructionist-inspired curricula [21] such as the Creative Computing Curriculum [6]. These were typified by open-ended, personally-meaningful projects that allowed students agency over what concepts they learned and applied, and to what they applied them. As the CSforAll movement has gained steam, there has been an increase in CS in formal elementary school settings. This has led to more scaffolded, structured approaches. For example, the Use->Modify->Create approach [18] supplements these open-ended (Create) projects with example code that more closely targets student learning to specific concepts (Use->Modify). Alternatively, with a puzzle-based curriculum such as Code.org, students complete many highly-structured small puzzles in order to practice individual skills in a single context. Once many of these skills are built, students may create a culminating project that allows them to apply those skills in their chosen context.

Other work has sought to situate CS instruction within the context of another core subject. Through a series of several youth programs, Lee et al. showed rich computational environments, i.e. environments that encourage access to the environment's internal mechanisms, allow students to create meaningful and instructive models of problems they have encountered in other areas [18]. Integration of CS education with pre-established disciplines, especially in STEM, is therefore an attractive option.

Prior work on integration has focused primarily on the middle school level or above. Zhang et al. have proposed and tested a framework for middle-school CS and science integration built around the Logic Programming language [41], and Rodger et al. have found success in introducing teachers, especially mathematics teachers, to the Alice programming environment [27]. In addition, Schanzer, Fisler et al. have shown that the integration of algebra and text-based programming in the Bootstrap curriculum leads to improvements in students' understanding of function composition and word problems, characterizing the effect as a transfer of skill supported by "deep structural connections between the domains" [31]. These approaches trace a pathway toward the widespread deployment of CS concepts in the classroom. We aim to expand on this work by exploring integration at the elementary level.

2.2 Learning Trajectories

Learning trajectories are hypothetical pathways that begin with students' prior knowledge, progress through instructional activities, and arrive at particular learning goals [35]. While actual routes to learning for particular students cannot be perfectly predicted [2], LTs are useful tools for guiding curriculum development because they provide research-based descriptions of how learning is likely to progress under ideal instructional conditions [7]. Further, LTs can serve as effective learning tools for teachers to develop models of how students might think about particular concepts [40]. Designing instructional materials with reference to LTs therefore has the potential to both guide coherent instructional experiences for students and support teachers' implementation of lessons featuring content where they are still developing expertise. Research is emerging about progressions and trajectories for CS learning [9, 32, 39, 25, 24, 23]. In this paper we build on this work by presenting a set of instruction materials developed with reference to CS LTs.

3 DESIGN PRINCIPLES

In this section, we discuss the six design principles (DPs) that guided the design of Action Fractions.

DP1: Mathematics Focus. We chose integration with mathematics because of the time dedicated to it during the typical elementary school day, the inclusion of all students in mathematics instruction, and synergies between mathematics and CS. To ensure mathematical coherence across our sequence of activities, we focus on one mathematical content domain: fractions, as covered in an existing mathematics curriculum.

We chose fractions for two reasons. First, there is a clear need for innovation in fraction instruction. Many students find the proportional reasoning involved in fractions difficult [5, 4] and struggle to overcome whole-number biases when working with fractions [36, 34]. Difficulties with fractions are particularly troublesome because fluency with fractions is critical for success in algebra and beyond [20]. Second, fractions are a good fit with several CS concepts. The utility of visual representations in learning fractions [17, 3] and the Common Core's [1] emphasis on visual representations fit well with the visualization inherent in computer programming at the elementary level.

DP2: Following CS LTs. We also needed tools for guiding a coherent sequence of CS content. As such, our second design principle was to follow a set of research-based CS LTs [26, 24, 23] to decide which CS ideas (e.g., repetition) to address in our lessons and in what order. For example, an early activity in our sequence, Fraction Circles 1, serves as an initial exposure to the effects of repeating the same command multiple times and how doing so can have a cumulative effect on the program output. A later activity builds on this idea and introduces the Scratch "repeat" block as a compact way to tell a computer to repeat commands.

DP3: Using a familiar lesson structure. Recognizing the challenges that both students and teachers would face in working through integrated lessons (often for the first time), we chose to structure the lessons to match the format of the companion mathematics curriculum. We felt the familiar structure would help both teachers and students to focus on the challenges of the new content. As such,

Figure 1: Fraction Circles 1 Lesson Opener

each integrated lesson features a warm-up activity, a focus activity that serves as the body of the lesson, and a wrap-up discussion. The lesson pages were also visually styled after the mathematics curriculum so teachers would know where to look for particular features or kinds of information. Figure 1 shows an example opening page of a lesson, which outlines the lesson activities.

DP4: Universal Design for Learning (UDL). In order to increase access and engagement, we planned instruction in a way that accounted for a wide range of learner differences by applying the principles of Universal Design for Learning (UDL [28]) to our lesson creation process. The UDL framework, developed by CAST (http://cast.org), has three primary principles: (1) Multiple means of engagement (the "why" of learning), (2) multiple means of representation (the "what" of learning), and (3) multiple means of expression (the "how" of learning). These principles provided a roadmap to design CS-integrated instruction based on students' potential strengths and challenges [16].

Through the iterative development process, lessons were reviewed to identify potential barriers to student learning and suggestions for teachers to help mitigate them. The nature of these barriers varied and included potential student challenges related to hardware or software use or related to academic content and language. Two other features designed to increase accessibility were included. First, "I Can" statements, were designed to make the learning goals explicit, activate prior knowledge, and allow students to self-monitor the learning processes and objectives. These "I Can" statements, written in student-friendly language, are introduced at the beginning of every lesson and revisited at the end of the lesson. During both lesson segments, students are asked to self-evaluate their understanding and mastery of the lesson goals. Second, another UDL-informed lesson feature focused on providing flexible options for adapting the lessons based on student preferences. These options might involve providing additional ways of engaging with the content with physical and virtual manipulatives or alternate ways in which students could show their learning. Figure 2 shows an example of all three UDL-informed features.

DP5: TIPP&SEE. We utilize a TIPP&SEE learning strategy[30] to teach students how to learn from the provided example code, mediated through a worksheet. It leads students in purposeful play

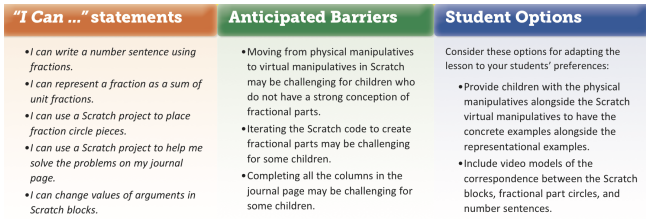

Figure 2: Fraction Circles 1 UDL Considerations

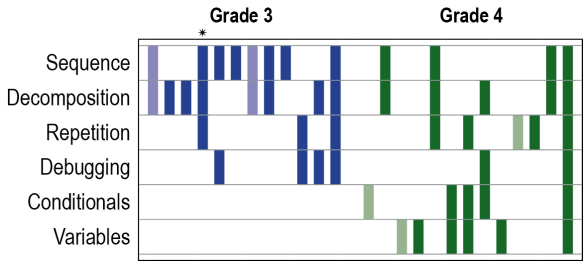

Figure 3: Lesson Sequence and CS Content Coverage. (Asterisk marks Fraction Circles 1.)

with recorded observation, finding and making predictions about code related to particular actions they observed, and deliberate tinkering by making code changes to understand how individual blocks work. Results have shown that students using TIPP&SEE perform better on post-assessments of loops questions[30], and they complete more technical requirements of the activities[13].

DP6: Comprehensive supports within lessons. As a final design principle, we strove to provide everything a teacher might need to implement the lessons within their classroom. Along with detailed descriptions of how to implement the lesson activities, our lessons include background information about the content for teachers, printable student pages, links to starter Scratch projects with custom blocks, discussion questions with sample student answers, slide decks, and when appropriate, suggestions for lesson adaptations. Each lesson opener page (Figure 1) also clearly identifies the mathematics content standards and target CS content to help teachers see how the lessons fit together. While such materials do not negate the need for teachers to carefully plan their teaching, we aimed to allow teachers to devote their planning time to understanding the lesson and its purpose rather than preparing materials.

4 DESCRIPTION OF MATERIALS

4.1 Lesson Sequence and CS Content Coverage

Figure 3 shows the full sequence of 25 integrated lessons and identifies the focus CS concepts in each—one or more of sequence, decomposition, repetition, conditionals, variables, or debugging. The light bars indicate unplugged activities. These Action Fractions lessons are interspersed throughout the main mathematics curriculum. At each grade level, during the three major chapters that cover fractions, they provide roughly weekly exposure to CS content.

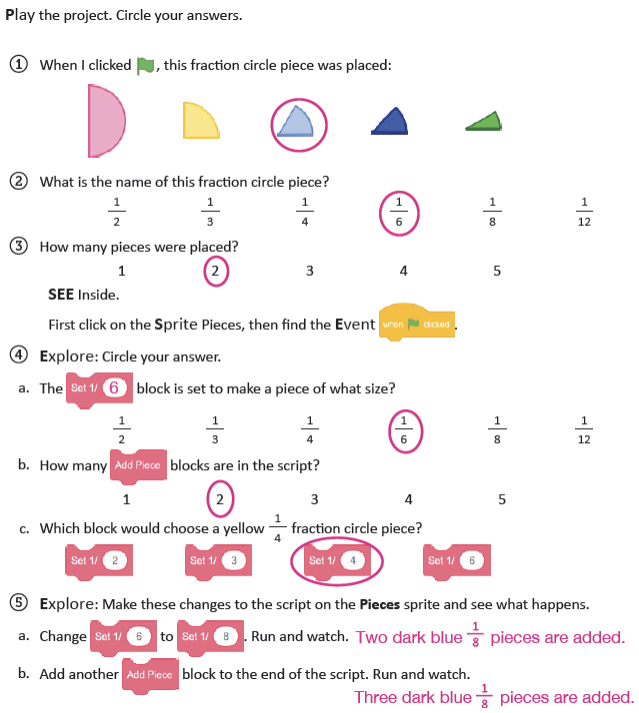

Figure 4: Students observe and explore the provided code.

4.2 Example Lesson: Fraction Circles 1

Fraction Circles 1 is an early 3rd grade lesson that begins with a whole-group warm-up using physical fraction circle piece manipulatives to focus students' attention on expressing fractions as sums of same-denominator unit fractions (e.g. $\frac{3}{8} = \frac{1}{8} + \frac{1}{8} + \frac{1}{8}$), the core mathematics idea for this activity. In the main activity, students interact with a pre-coded Scratch project. Custom blocks are provided to control which size fraction-circle piece will be shown in the virtual manipulative and place these pieces. Students' introduction to the code blocks and their operations is scaffolded with a TIPP&SEE worksheet (Figure 4). The goal is for them to understand that they can control what size piece is shown (the denominator) and how many pieces are shown (the numerator) on the virtual manipulative by making changes to the code. A teacher-led whole-group discussion reviews what they learned about the project. Students then complete a journal page with a set of exercises, each with a target fraction (Figure 5). Students write the fraction as a number, draw the fraction-circle picture, modify the script, and write a number sentence that shows the fraction as a sum of unit fractions. As noted by the asterisk in Figure 3, the CS focus areas include sequence (students must use specific instructions that are understood by the computer), repetition (students use the same Add Piece block repeatedly and observe the cumulative effect), and decomposition (students discuss first setting the denominator, then adding pieces).

5 PILOT IMPLEMENTATION

Here we describe our pilot of the Action Fractions materials, including the participants, data, analysis, and key results.

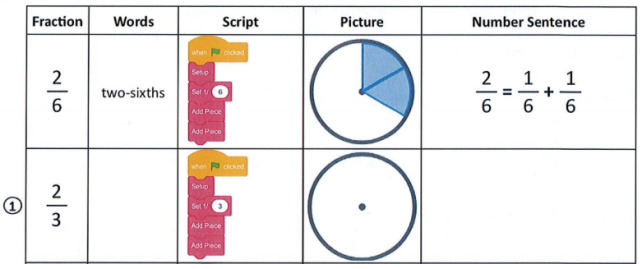

Figure 5: Part of a student journal page. Each exercise has some fields filled in; students fill in the other fields.

5.1 Participants and Recruitment

Elementary teachers were recruited from schools using the companion math curriculum. Our teacher participants were generalists, with no background in CS. All participating teachers attended a one or two-day professional development workshop (PD) to introduce them to the Scratch environment and Action Fractions materials. In the 2018–19 school year, five teachers, six classes, and 105 students participated, and in the 2019–20 school year, eleven teachers, twelve classes, and 257 students piloted our lessons.

5.2 Data Collection

Four data sources were collected and analyzed: (a) TIPP&SEE student worksheets (described above), (b) student journal pages where they recorded the answers to the exercises with target fractions, (c) feedback surveys that teachers completed after implementing each lesson, and (d) teacher interviews. Student work was collected, de-identified, scanned, and stored on a secure cloud server. Teacher feedback surveys were completed online. Teacher interviews were conducted and transcribed at the end of each academic year.

During recruitment and PD, teachers expressed general concerns about the required CS expertise, the amount of curricular time the lessons would require, the level of engagement of the materials (beyond the novelty of the "coding"), and the relevance of the CS content with respect to their core mathematics curriculum. We surmised that these concerns were issues that would not only impact their practice, but how they spoke to administrators, parents, and students about the study. Accordingly, we attended to these specific issues when designing our teacher surveys and interviews.

5.3 Analysis

We collated teachers' responses to the feedback surveys to gain a sense of how accessible and engaging they found the lessons, the time it took to implement them, and any other issues related to their concerns during recruitment and PD. To supplement the overall picture of teachers' responses to the lessons, we looked through the open-response sections of the surveys and the interview transcripts to support or further explain the trends. We also looked for quotations that elucidated teachers' reactions to how we operationalized specific design principles.

Student TIPP&SEE and journal page responses were coded and analyzed for accuracy and completeness. We then reviewed the

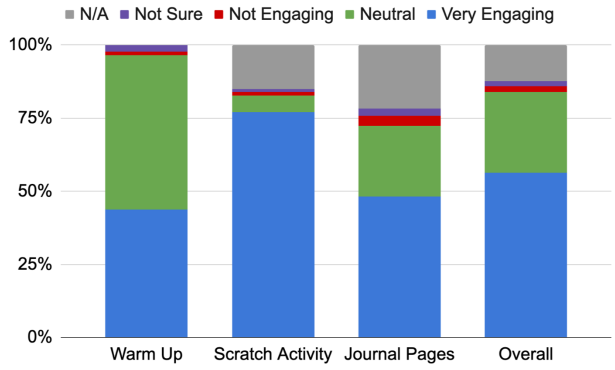

Figure 6: Teachers rated student engagement for various lesson components.

interesting unexpected or incorrect answers in more detail to determine what types of errors students made and infer reasons why they may have produced the unexpected answers. In Section 5.4.2 we include one set of interesting student answers.

5.4 Results

We present two sets of results. First, we discuss the extent to which we were able to operationalize our design principles to create lessons that were usable, enjoyable, and accessible to students and teachers. Second, we present an interesting pattern of results in the student data about how students navigated the constraints the lesson placed on both the computational and mathematical aspects of their solutions.

5.4.1 Operationalization of Design Principles. The six design principles laid out in Section 3 represent our best intentions to address barriers that generalist elementary school teachers may face as they attempt to implement integrated CS curriculum. Here we present results as to how well our lessons addressed their particular concerns about student engagement, instructional time, and relevance to their mathematics curriculum.

Figure 6 shows that in general, teachers highly rated the level of student engagement across the lessons. For example, in Fraction Circles 1, one teacher said, "I felt students were focused and really in tune to what they were learning." With respect to time, 72% of responses on the feedback survey indicated the lessons were an appropriate length. Four out of five teachers responded that they completed Fraction Circles 1 in the recommended 60 minutes of instructional time. However, even though most teachers rated the time required as appropriate, some still commented that they needed 15 to 20 extra minutes for students to complete the activities and the whole-class wrap-up discussions. Several teachers chose to dedicate additional instructional time to complete these lessons when needed, which further supports their assessment of the activities as engaging for students.

The teacher surveys and interviews provide additional evidence of the role our design principles played in the success of the lessons. Teachers appreciated the ways in which the lessons attended to the new CS learning and enriched the math learning. One teacher commented that Action Fractions lessons, "not only added another component in a fun way to learn what they were already learning in class but then it did deepen their understanding." Teachers also specifically mentioned the UDL features in their end-of-year interviews. One teacher said, "I enjoyed the I Can statements at the beginning because a lot of them [students] gave thumbs down on everything and then by the end, you know, it made sense [to them]." Teachers also responded well to familiar lesson structure, with one saying "[I] like [how] the teacher lesson guide mimics ... the actual math lesson content."

5.4.2 Constraint and Variation. The Action Fractions lessons make ample use of custom blocks to allow students to express the mathematics directly as code and to show the effects visually. For example, in Fraction Circles 1, the mathematical focus is building non-unit fractions from unit-fraction pieces. To allow students to express this mathematical idea directly in code, we provided custom blocks to set the denominator and place individual unit-fraction pieces. We used custom blocks in part to facilitate tight coupling to the mathematics concepts (see DP1). We also felt that using custom blocks often highlighted a particular idea from the Sequence LT (see DP2), which is that programs are built from a limited set of instructions. The TIPP&SEE worksheets (see DP5) allowed us to guide students in discovering what the custom blocks did.

Despite the advantages of using custom blocks in relation to our DPs, reliance on custom blocks did introduce a tension in terms of the variety of solutions students would produce. The specificity of the custom blocks placed limitations on the sequences of blocks that could produce a particular answer, especially if students only used the blocks that had been formally introduced. In the case of Fraction Circles 1, for example, the top row in Figure 5 shows the only code sequence that would show $\frac{2}{6}$ using the blocks students had explored. We wondered if such constraints might limit the variety of solutions students produced.

As it turned out, students found ways to create varied solutions even under the restrictions placed on their code. Our implementation data showed that while many students produced the expected solutions using the custom blocks, it was not uncommon for students to produce answers to some parts of the problems that showed richer variation. For example, in the case of Fraction Circles 1, four third graders used the Repeat block in their code instead of adding multiple Add Piece blocks. One example is shown at the top of Figure 7. This occurred despite our deliberate choice not to introduce the repeat block in this lesson and instead allow students to explore how adding multiple copies of the same block resulted in a cumulative effect on the output (see discussion of DP2). Three additional third graders used multiplication symbols in their representations of the code to indicate multiple repeat blocks (as in the bottom of Figure 7), suggesting they were thinking about how to condense their code even if they had not yet discovered the repeat block.

More variation was evident in some students' numbers sentences: students recorded mathematical representations of their solutions that showed uses of mathematics other than the intended addition of unit fractions. Eleven students used addition with non-unit fractions (e.g., $\frac{2}{12} + \frac{2}{12} = \frac{4}{12}$) and seven used addition with zero (e.g., $\frac{3}{4} + \frac{0}{4} = \frac{3}{4}$). We found this result particularly interesting because all of these are correct and display mathematical understanding, but they do not correspond with the blocks available for the script.

Figure 7: Student work featuring the use of repetition.

For example, there is no block corresponding to a non-unit fraction. Thus, these students displayed some variation in their mathematical solutions but were not able to express those solutions with the code blocks we provided.

Responses outside those suggested by the custom blocks also occurred in other lessons. For example, in a lesson about addition of fractions, students sometimes produced solutions other than what we intended (for example, producing the number sentence $\frac{2}{3} + \frac{0}{3} = \frac{2}{3}$ when asked to find two fractions that sum to $\frac{2}{3}$, when our expectation was $\frac{1}{3} + \frac{1}{3} = \frac{2}{3}$). The custom blocks in this lesson would allow students to create code with a 0 in the numerator. However, students did not usually record scripts on their journal pages that matched these mathematically varied solutions.

6 DISCUSSION

Here we examine insights from our design process and implementation results and discuss the successes and challenges they highlight.

6.1 Successes

Overall, our six design principles were utilized to create a coherent set of integrated mathematics+CS lessons. Many of these lessons were manageable for teachers to implement in a class period, and teachers appreciated many of the features that were embedded in the lessons (Section 5.4). Thus, this serves as a proof-of-concept that the creation of integrated lessons that target important mathematical and computer-science content—and that are feasible for elementary teachers to implement—is possible. Future research is needed to investigate the impact of such lessons on students' attitudes towards and learning of both disciplines.

The variation displayed by some students in their responses, perhaps constrained by the custom blocks and structure of the projects, eased concerns about the limitations of our lessons. In particular, the handful of students using the repeat block prior to its introduction illustrated that even within lessons constrained by mathematics content, Scratch retained its "high-ceiling" quality. Moreover, the mathematical limitations on what the custom blocks were able to represent did inhibit students from thinking about other ways of creating their number sentences.

6.2 Challenges

Although we were pleased to find that students and teachers responded positively to most lessons, we would be remiss to not acknowledge the difficulties involved in creating these lessons. Each lesson required significant collaboration between expert CS education researchers, mathematics education researchers, curriculum developers, and current and former classroom teachers. The fully developed lessons are the result of several years of discussion and collaboration. Even so, some teacher feedback indicates certain lessons had a high level of difficulty or could not be feasibly implemented in a traditional class period. The overall challenge of creating coherent, integrated CS materials highlights the importance of supporting teachers by providing such materials—or at the very least, the backbones of such materials. Full creation of integrated materials is a heavy burden to place on teachers, even when those teachers have adequate CS knowledge.

The integration context significantly constrained what and how much CS content covered, resulted in unequal coverage both between CS content areas (Figure 3) and within CS content areas (multiple activities focus on a single point in the LT). For example, Decomposition is in more than half of the lessons, but most of those lessons are addressing the same big idea in the LT (Problem decomposition is a useful early step in problem solving).

The overall impact of these challenges is dependent on the purposes that integrated activities such as ours are meant to serve. Although our presented data did not focus on this, we believe our activities have the potential to support students in using computing to enrich their mathematical learning and gaining familiarity with some basic computing concepts. However, the activities do not serve as a comprehensive introduction to beginning computing.

When it comes to the variation students displayed within their responses, while we were impressed by the student ingenuity, we also acknowledge that such responses raise questions about whether the activities adequately support students in seeing connections between their pictures, fractions, number sentences, and code. When students used numerators or operators with no equivalents in the provided code blocks, they may not have seen these connections.

This reveals a tension in the activity design. On one hand, it is possible to create custom blocks that would support more variety in the student solutions. For example, in the Fraction Circles 1 activity, if we wanted to support the expression of + $\frac{2}{12}$, we could add an input to the Add Piece block for how many pieces to place. Then a single instruction would place multiple pieces, equivalent to a non-unit fraction. This might allow more students to see the connections between their pictures, fractions, number sentences, and code. However, this change has the potential to pull focus from the intended mathematics learning goal, which is for students to see that every fraction as composed of unit fractions. In addition, the more flexible block is more complex, which could be inappropriate for such an early computing activity. In general, we faced challenges in balancing a desire to support creative variation in student responses and target specific learning goals.

7 CONCLUSION

The Action Fractions instructional materials and pilot implementation results illustrate both the possibility of creating coherent integrated mathematics and CS materials and the many challenges involved in doing so. We hope that our design principles support future efforts in integrated materials development and our discussion of challenges starts a conversation about how to best distribute this ambitious work among researchers, developers, and teachers.

8 ACKNOWLEDGMENTS

We are grateful to the teachers and students who piloted Action Fractions in their classrooms with such dedication and enthusiasm. This project was funded by National Science Foundation (NSF) Grant No. 1932920.

REFERENCES

[1] National Governors Association et al. "Common core state standards". In: *Washington, DC* (2010).

[2] Michael T Battista. "Conceptualizations and issues related to learning progressions, learning trajectories, and levels of sophistication". In: *The Mathematics Enthusiast* 8.3 (2011), pp. 507–570.

[3] Merlyn J Behr et al. *Rational number concepts*. Ed. by R. Lesh and M. Landau. Academic Press, pp. 91–126.

[4] Ty W Boyer and Susan C Levine. "Child proportional scaling: Is 1/3= 2/6= 3/9= 4/12?" In: *Journal of Experimental Child Psychology* 111.3 (2012), pp. 516–533.

[5] Ty W Boyer, Susan C Levine, and Janellen Huttenlocher. "Development of proportional reasoning: Where young children go wrong." In: *Developmental psychology* 44.5 (2008), p. 1478.

[6] Karen Brennan, Christian Balch, and Michelle Chung. "Creative computing". In: *Harvard Graduate School of Education* (2014).

[7] Jere Confrey, Alan P Maloney, and Andrew K Corley. "Learning trajectories: A framework for connecting standards with curriculum". In: *ZDM* 46.5 (2014), pp. 719–733.

[8] Stephen Cooper, Wanda Dann, and Randy Pausch. "Alice: A 3-D Tool for Introductory Programming Concepts". In: *Proceedings of the Fifth Annual CCSC Northeastern Conference on The Journal of Computing in Small Colleges*. CCSC '00. Ramapo College of New Jersey, Mahwah, New Jersey, USA: Consortium for Computing Sciences in Colleges, 2000, 107–116.

[9] Hilary Dwyer et al. "Identifying elementary students' pre-instructional ability to develop algorithms and step-by-step instructions". In: *Proceedings of the 45th ACM technical symposium on Computer science education*. 2014, pp. 511–516.

[10] Georgios Fessakis, Evangelia Gouli, and E Mavroudi. "Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study". In: *Computers & Education* 63 (2013), pp. 87–97.

[11] Louise P Flannery et al. "Designing ScratchJr: support for early childhood learning through computer programming". In: *Proceedings of the 12th International Conference on Interaction Design and Children*. ACM. 2013, pp. 1–10.

[12] Krista Francis and Brent Davis. "Coding robots as a source of instantiations for arithmetic". In: *Digital Experiences in Mathematics Education* 4.2-3 (2018), pp. 71–86.

[13] Diana Franklin et al. "Exploring Student Behavior Using the TIPP&SEE Learning Strategy". In: *Proceedings of the 2020 ACM Conference on International Computing Education Research*. ICER '20. Virtual Event, New Zealand: Association for Computing Machinery, 2020, 91–101. ISBN: 9781450370929. DOI: 10.1145/3372782.3406257. URL: https://doi.org/10.1145/3372782.3406257.

[14] Maya Israel and Todd Lash. "From classroom lessons to exploratory learning progressions: mathematics+ computational thinking". In: *Interactive Learning Environments* 28.3 (2020), pp. 362–382.

[15] Maya Israel et al. "Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis". In: *Computers and Education* 82 (2015), pp. 263–279. DOI: https://doi.org/10.1016/j.compedu.2014.11.022.

[16] Maya Israel et al. "Teaching Elementary Computer Science through Universal Design for Learning". In: *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. 2020, pp. 1220–1226.

[17] Nancy C Jordan et al. "Developmental predictors of fraction concepts and procedures". In: *Journal of Experimental Child Psychology* 116.1 (2013), pp. 45–58.

[18] Irene Lee et al. "Computational Thinking for Youth in Practice". In: *ACM Inroads* 2.1 (Feb. 2011), pp. 32–37. ISSN: 2153-2184. DOI: 10.1145/1929887.1929902. URL: http://doi.acm.org/10.1145/1929887.1929902.

[19] John Maloney et al. "The Scratch Programming Language and Environment". In: *Trans. Comput. Educ.* 10.4 (Nov. 2010), 16:1–16:15. ISSN: 1946-6226. DOI: 10.1145/1868358.1868363. URL: http://doi.acm.org/10.1145/1868358.1868363.

[20] National Mathematics Advisory Panel. *Foundations for success: The final report of the National Mathematics Advisory Panel*. US Department of Education, 2008.

[21] Seymour Papert. *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc., 1980.

[22] Christina Pei, David Weintrop, and Uri Wilensky. "Cultivating computational thinking practices and mathematical habits of mind in lattice land". In: *Mathematical Thinking and Learning* 20.1 (2018), pp. 75–89.

[23] Kathryn M. Rich et al. "A K-8 Debugging Learning Trajectory Derived from Research Literature". In: *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. SIGCSE '19. Minneapolis, MN, USA: ACM, 2019, pp. 745–751. ISBN: 978-1-4503-5890-3. DOI: 10.1145/3287324.3287396. URL: http://doi.acm.org/10.1145/3287324.3287396.

[24] Kathryn M. Rich et al. "Decomposition: A K-8 Computational Thinking Learning Trajectory". In: *Proceedings of the 2018 ACM Conference on International Computing Education Research*. ICER '18. Espoo, Finland: ACM, 2018, pp. 124–132. ISBN: 978-1-4503-5628-2. DOI: 10.1145/3230977.3230979. URL: http://doi.acm.org/10.1145/3230977.3230979.

[25] Kathryn M. Rich et al. "K-8 Learning Trajectories Derived from Research Literature: Sequence, Repetition, Conditionals". In: *Proceedings of the 2017 ACM Conference on International Computing Education Research*. ICER '17. Tacoma, Washington, USA: ACM, 2017, pp. 182–190. ISBN: 978-1-4503-4968-0. DOI: 10.1145/3105726.3106166. URL: http://doi.acm.org/10.1145/3105726.3106166.

[26] Kathryn M Rich et al. "K-8 learning trajectories derived from research literature: Sequence, repetition, conditionals". In: *Proceedings of the 2017 ACM Conference on International Computing Education Research*. ACM. 2017, pp. 182–190.

[27] Susan Rodger et al. "Integrating Computing into Middle School Disciplines Through Projects". In: *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*. SIGCSE '12. Raleigh, North Carolina, USA: ACM, 2012, pp. 421–426. ISBN: 978-1-4503-1098-7. DOI: 10.1145/2157136.2157262. URL: http://doi.acm.org/10.1145/2157136.2157262.

[28] David H Rose and Anne Meyer. *Teaching every student in the digital age: Universal design for learning*. ERIC, 2002.

[29] José-Manuel Sáez-López, Marcos Román-González, and Esteban Vázquez-Cano. "Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools". In: *Computers & Education* 97 (2016), pp. 129–141.

[30] Jean Salac et al. "TIPP&SEE: A Learning Strategy to Guide Students through Use–>Modify Scratch Activities". In: *Proceedings of the 2019 ACM SIGCSE Technical Symposium on Computer Science Education*. SIGCSE '19. ACM. Portland, OR, USA, 2019.

[31] Emmanuel Schanzer et al. "Transferring Skills at Solving Word Problems from Computing to Algebra Through Bootstrap". In: *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. SIGCSE '15. Kansas City, Missouri, USA: ACM, 2015, pp. 616–621. ISBN: 978-1-4503-2966-8. DOI: 10.1145/2676723.2677238. URL: http://doi.acm.org/10.1145/2676723.2677238.

[32] Linda Seiter and Brendan Foreman. "Modeling the learning progressions of computational thinking of primary grade students". In: *Proceedings of the ninth annual international ACM conference on International computing education research*. 2013, pp. 59–66.

[33] Pratim Sengupta, Amanda Dickes, and Amy Farris. "Toward a phenomenology of computational thinking in STEM education". In: *Computational thinking in the STEM disciplines*. Springer, 2018, pp. 49–72.

[34] Robert S Siegler et al. "Fractions: The new frontier for theories of numerical development". In: *Trends in cognitive sciences* 17.1 (2013), pp. 13–19.

[35] Martin A Simon. "Reconstructing mathematics pedagogy from a constructivist perspective". In: *Journal for research in mathematics education* (1995), pp. 114–145.

[36] Stamatia Stafylidou and Stella Vosniadou. "The development of students' understanding of the numerical value of fractions". In: *Learning and instruction* 14.5 (2004), pp. 503–518.

[37] Rivka Taub, Michal Armoni, and Mordechai Ben-Ari. "CS unplugged and middle-school students' views, attitudes, and intentions regarding CS". In: *ACM Transactions on Computing Education (TOCE)* 12.2 (2012), pp. 1–29.

[38] David Weintrop et al. "Defining computational thinking for mathematics and science classrooms". In: *Journal of Science Education and Technology* 25.1 (2016), pp. 127–147.

[39] Linda Werner et al. "The fairy performance assessment: measuring computational thinking in middle school". In: *Proceedings of the 43rd ACM technical symposium on Computer Science Education*. 2012, pp. 215–220.

[40] P Holt Wilson, Gemma F Mojica, and Jere Confrey. "Learning trajectories in teacher education: Supporting teachers' understandings of students' mathematical thinking". In: *The Journal of Mathematical Behavior* 32.2 (2013), pp. 103–121.

[41] Yuanlin Zhang et al. "LP Based Integration of Computing and Science Education in Middle Schools". In: *Proceedings of the ACM Conference on Global Computing Education*. CompEd '19. Chengdu,Sichuan, China: ACM, 2019, pp. 44–50. ISBN: 978-1-4503-6259-7. DOI: 10.1145/3300115.3309512. URL: http://doi.acm.org/10.1145/3300115.3309512.