Framing Computational Thinking for Computational Literacies in K-12 Education

Kafai, Yasmin B.; Proctor, Chris; Lui, Debora A.

PROCEEDINGS OF THE WEIZENBAUM CONFERENCE 2019
CHALLENGES OF DIGITAL INEQUALITY
DIGITAL EDUCATION | DIGITAL WORK | DIGITAL LIFE

FRAMING COMPUTATIONAL THINKING FOR COMPUTATIONAL LITERACIES IN K-12 EDUCATION

Yasmin B. Kafai
University of Pennsylvania
Philadelphia, PA USA
kafai@upenn.edu

Chris Proctor
Stanford University
Palo Alto, CA USA
cproctor@stanford.edu

Debora A. Lui
University of Pennsylvania
Philadelphia, PA USA
deblui@upenn.edu

ABSTRACT

The last decade has seen an increased interest in promoting computing education for all, focused on the idea of "computational thinking." Currently, three framings for promoting computational thinking in K-12 education have been proposed, emphasizing either (1) skill and competency building, (2) creative expression and participation, or (3) social justice and reflection. While each of these emphases is valuable and needed, their narrow focus can obscure important issues and miss critical transformational opportunities for empowering students as competent, creative, and critical agents. We argue that these computational framings should be seen as literacies, thereby historicizing and situating computer science with respect to broader educational concerns and providing new directions for how schools can help students to actively participate in designing their digital futures.

KEYWORDS

Computational thinking; literacy; critical pedagogy; programming

DOI: 10.34669/wi.cp/2.21

1 INTRODUCTION

The growing pervasiveness of digital technologies has promoted a global interest in computer science education for all. In addition to expanding access to specialized coursework in computer science, there has been an effort to define a core set of computational skills that every child should learn and use across the curriculum, as well as in everyday life. Efforts to define such a broadly-applicable yet distinct skillset have coalesced around *computational thinking*, a term promoted by Wing (2006) naming the foundational skills and thought processes of computer science. Learning computational thinking is being equated in many cases with learning the language of computers for digital literacy in the 21st century.

Since Wing's publication, numerous papers and books (e.g., Rich & Hodges, 2017), and hundreds of studies (e.g., Florez et al., 2017) have been dedicated to investigating student learning, teaching approaches, and assessments of computational thinking for K-12 education. While there is no agreed upon definition of computational thinking (National Research Council, 2010), it is nonetheless being proposed as a foundation for new sets of 'literacies' focused on computational understanding and skill (diSessa, 2001; Guzdial, 2019; Vee, 2017).

In this paper, we describe three different framings for how computational thinking has been conceptualized and review their strengths and weaknesses. We then address missing aspects in each of these framings and discuss how our analysis of computational thinking can provide a rationale for a broader conception of computational literacies in K-12 CS education.

material.

2 THREE FRAMINGS OF COMPUTATIONAL THINKING

Wing (2006) defined computational thinking most generally as "taking an approach to solving problems, designing systems and understanding human behavior that draws on concepts fundamental to computing" (p. 33). Early efforts in the 1980s of bringing computers and programming into schools most often presented it as a standalone activity, while today's focus on computational thinking is an interdisciplinary practice, particularly relevant to STEM fields (Weintrop, Beheshti, Horn, Orton, Jona, Trouille & Wilensky, 2016). We have identified three directions that have framed computational thinking from different learning perspectives.

The first, and most prominent, direction under which computational thinking has been framed comes from a functional, cognitive perspective, which supports the pragmatic goals of skill and competence building. This direction draws from cognitive research traditions that already dominated efforts to introduce programming in the 1980s (Soloway & Spohrer, 1989) which place computational thinking as a form of complex problem solving that is primarily performed by individuals (Grover & Pea, 2013). Here, emphasis is placed on student learning of computational concepts such as loops, recursion, conditionals, and data structures, and practices such as iteration and abstraction, which are in turn connected to future opportunities for work and career advancement. The cognitive framing is most prevalent in current national frameworks and curricula such as Code.org's *CS Discoveries* and *Explorations* that outline learning pathways for K-12 students.

Another direction emphasizes how students can develop computational thinking through designing and programming shareable digital artifacts. This direction draws from constructionist learning theory (Papert, 1980), which emphasizes interest-driven and peer-supported activities and thus sees computational thinking as a vehicle for personal expression and participation (Kafai & Burke, 2014). Learning key computational concepts and practices are thus situated within acts of designing complex applications that are shared on social networks. Much work following this direction has studied youth' engagement with computer science activities within informal, non-school, education environments such as community technology centers and online communities. However, there have also been several efforts aimed at developing school curriculum using this situated approach, including the *Creative Computing Guide* that highlights game design and storytelling activities (Brennan, Balch & Chung, 2019) and the *Stitching the Loop* activities that engage students in crafting and coding personalized electronic textiles (Kafai & Fields, 2018).

Finally, a third direction places students' computational thinking into the context of social justice-oriented production in order to engage them with existing socio-political issues. This framing draws heavily from the traditions of critical pedagogy, which emphasizes both an examination of and resistance to oppressive power structures (Freire, 1993) as well as production-oriented media literacy, which develops youth agency through the process of creating and disseminating media content (Buckingham, 2003). Efforts following this direction place computational thinking as a platform through which to address existing real-world challenges by creating original multimedia artifacts. Student-generated projects include, for example, digital mapping visualizations that highlight local issues with gentrification occurring (Lee & Soep, 2016), and mobile apps that challenge existing narratives about 'low-resource' neighborhoods through highlighting accessible extracurricular activities that students have catalogued for their peers (Vakil, 2018).

One striking commonality in all three approaches is that today's computational thinking is situated in the design and production of authentic, real-world digital applications, very much in contrast to the short, isolated programming activities that dominated CS teaching in the 1980s (Palumbo, 1980). In the skill and competency building approach, students engage in the production of digital objects such as video games and robots for the purposes of supporting understanding of fundamental CS concepts such as rule-based behaviors and abstraction (Grover & Pea, 2013). Approaches that favor personal and creative expression explicitly connect to learners' interests in digital media by privileging personalized artifacts that can be shared with others, often referencing popular culture themes (Richard & Kafai, 2016). Finally, in social justice-oriented design students consider design applications to address and critique existing inequities and oppressions through their designs (Lee & Soep, 2016).

Not surprisingly these three framings differ in how they balance explicit efforts to build individual skill and disciplinary knowledge with efforts to engage students in applying these for personally meaningful purposes, whether creative or political. We know from studies of Scratch online community that creative expressions and participation alone are not sufficient to make computational concepts and practices accessible to novice programmers (Kafai & Burke, 2014). Relatedly, it is not clear from project descriptions in social justice-focused projects (Soep & Lee, 2016) what students actually learned in terms of computational concepts and practices (one exception is Lee & Garcia, 2015), something which highlights a greater focus on content creation, rather than understanding the actual mechanisms of computational infrastructures, which are often themselves a source of oppression and inequality (Vakil, 2018).

3 TOWARDS COMPUTATIONAL LITERACIES

While oftentimes these framings are cast against one another (see Vakil, 2018), we argue that they are indeed complementary and necessary to support students' development as computationally engaged agents in the complex, digital world in which they participate. For that reason, we propose that all three framings of computational thinking become the foundation for computational literacies (diSessa, 2001; Guzdial, 2019; Vee, 2017). The functional framing recognizes the infrastructural role of computers in literacy practices (diSessa, 2001) and emphasizes their skillful use. The situated framing recognizes how computers mediate social identities and meaning-making (Vee, 2017). The critical framing continues the historical analysis of the relationship between media and power (Vakil, 2018).

Integrating these three framings in a pragmatic, situated, and critical vision of computational literacies is useful as a design heuristic, surfacing missed opportunities in existing curricular initiatives. Functionally-oriented curricula are at risk of (ironically) rendering computational thinking as a form of 'book learning' disconnected from students' identities and lived experience. Situated framings sometimes emphasize creativity and self-expression without substantially developing students' skills or understanding of the computational media they use. Additionally, situated framings rarely consider how the actual mechanics of how popularity is measured online or the digital infrastructures through which content circulation occurs.

On the surface, social justice-oriented framings of computational thinking claim to move from an individualistic perspective that only examines personal choice to understanding the larger socio-political issues that frame problematic uses of technology including data mining and tracking, targeted marketing, and privacy issues around surveillance and citizen's rights. But a closer examination of actual implementations in this area reveals an orientation toward developing production-oriented skill building and emphasis on content creation, rather than deep analysis of the structures of computation itself. From this perspective, students are missing out on developing a critical understanding of what computation is in our world today.

The recent emergence of critical issues around the circulation of fake news, privacy violations, and algorithmic bias make clear that students need to know not only how to use and design digital media—promoted in the current framings of computational thinking—but also to question the design, infrastructures, and histories of digital technologies themselves. Individually, the proposed application designs in the distinct framings of computational thinking fall short of not only getting youth to review and understand the existing computational mechanisms that underlie digital engagement, but also in considering the role of computation can play in both supporting and suppressing individual and social self-determination online. While others (Vakil, 2018) have already outlined the necessity of engaging youth in the critical engagement of computational infrastructure, we additionally highlight the need to combine this view (which views computation as something which can take away power and agency) with the situated, constructionist framing (which emphasizes computation as something that can also grant power and agency). In other words, any emphasis on computational literacies should equally consider not only what competencies students can acquire, but also how to create awareness of the ways the use and design of computational media can simultaneous oppress and inspire.

We close this section with an example that promotes such an integrated and situated approach toward computational literacies. One important idea to consider here are *where* and *how* the actual intersections between these frameworks occur (e.g., skills building, creative expression, and social justice) and how we can further push the integration between these perspectives, while also highlighting how they are situated in existing contexts. The example of visualizing and questioning computational participation in Scratch illustrates how computational thinking can be reframed as integrated, situated computational literacies. Scratch is an online programming community, which has attracted over ten million of kids in the creation of programming projects that are shared online. Studies have been conducted to understand what computational concepts and practices Scratch users engage with, the variety of creative projects, and the progression of computational participation for different types of users (Kafai & Burke, 2014).

However, more recently, researchers have added new programming features, special "community blocks" that let Scratch users not just create projects but also help them understand how participation data is collected, used and disseminated on the site, and also within many existing online communities (Dasgupta & Hill, 2017). By giving users opportunity to creatively play and develop personalized projects with these blocks, students became more cognizant of numerous issues surrounding big data today, whether a realization of the privacy implication of data collection and retention, possible avenues for exclusion through data-driven algorithms, or possible biases and assumptions hidden within supposedly neutral data claims. While the users only expressed these perspectives on the closed system of Scratch (Hautea, Dasgupta & Hill, 2017), one can see how an understanding of these ideas can help promote a larger understanding of our wider digital environment today, where algorithms and data structures are rarely shared. Here, learning computational concepts is not just an instrumental goal but pushes youth towards considering the larger socio-political implications of data collection, analysis and use, thereby working to bring together the more cognitive, functional framing of CT with approaches that favor critical engagement and action.

This is one example of how the different framings of computational thinking could be integrated in computational literacies. As Scribner (1984) argued, literacy is not just about the pragmatics of reading and writing text but also about understanding their personal and political dimensions of texts. From this vantage point, computational literacies point towards three important dimensions of relationships that computational thinking needs to help realize: a pragmatic view about interacting with computational artifacts, programming and understanding code, a personal view about authoring one's identities and a political view about using, understanding, and transforming sense-making processes and subject positions. If computational literacies are to be included in the canon of K-12 literacies, then we need to move any single framing that only either emphasizes skill building or contextual uses, toward a larger view that embraces a need to combine these perspectives in order to highlight the values, biases, and histories embedded in the digital technologies.

4 CONCLUSIONS

In this paper, we outlined different framings for computational thinking and discussed their strengths and weaknesses. We proposed a new direction to reframe computational thinking as situated, critical literacies that not only encompasses functional skills, but also the socio-political and personal contexts that inherently accompany youth's use and production of digital media. Advancing computational literacies which situate learning and teaching of computer science education aligns with other efforts in K-12 education have as its overarching goal to educate students as responsible citizens—consumers, producers and critics—in the digital publics they all participate in and contribute to.

5 ACKNOWLEDGMENTS

The writing of this paper was supported by a CS-ER grant from Google to Yasmin Kafai. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of Google, the University of Pennsylvania, or Stanford University.

6 REFERENCES

1. Brennan, K., Balch, C., & Chung, M. & (2019). Creative Computing 3.0. Cambridge, MA: Harvard University.
2. Buckingham, D. (2003). Media education: Literacy, learning and contemporary culture. Cambridge, UK: Polity Press.
3. Dasgupta, S. & Hill, B. M. (2017). Scratch Community Blocks: Supporting Children as Data Scientists. In Proceedings of the 2017 CHI Conference on

Human Factors in Computing Systems (CHI '17) (pp. 3620-3631). New York, NY: ACM.

4. diSessa, A. (2001). Changing minds: Computers, learning, and literacy. Cambridge, MA: The MIT Press.
5. Florez, F. B., Casallas, R., Hernández, M., Reyes, A., Restrepo, S. & Danies, G. (2017). Changing a generations' way of thinking: Teaching computational thinking through programming. Review of Educational Research, 87(4), 834-860.
6. Freire, P. (1993). Pedagogy of the oppressed (20th anniversary edition). New York, NY: Continuum.
7. Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. Educational Researcher, 42(2), 59-69.
8. Guzdial, M. (2019). Computing Education as a Foundation for 21st Century Literacy. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19) (pp. 502-503.). New York, NY: ACM.
9. Hautea, S., Dasgupta, S., & Hill, B. M. (2017, April). Youth perspectives on critical data literacies. In Proceedings of the CHI'17 conference (pp. 919-930). New York, NY: Association for Computing Machinery.
10. Kafai, Y. B. & Burke, Q. (2014). Connected Code: Why children need to learn programming. Cambridge, MA: The MIT Press.
11. Kafai, Y. B., & Fields, D. A. (2018, August). Some reflections on designing constructionist activities for classrooms. In V. Dagiene & E. Jastuė, Constructionism 2018: Constructionism, Computational Thinking and Educational Innovation: conference proceedings, Vilnius, Lithuania (pp. 606-612). Available at http://www.constructionism2018.fsf.vu.lt/proceedings
12. Lee, C. H., & Garcia, A. D. (2014). "I want them to feel the fear …" Leveraging critical computational literacies for English Language Arts success. In R. E. Ferdig, &K. E. Pytash (Eds.), Exploring multimodal composition and digital writing (pp. 364–378). Hershey, PA: Information Science Reference.
13. Lee, C. H., & Soep, E. (2016). None but ourselves can free our minds: Critical computational literacy as a pedagogy of resistance. Equity & Excellence in Education, 49(4), 480-492.
14. National Research Council (2010). Report of a workshop on the scope and nature of computational thinking. Computer Science and Telecommunications Board (CSTB). Washington, DC: National Academy Press.
15. Palumbo, D. (1990). Programming Language/Problem-solving Research: A Review of Relevant Issues. Review of Educational Research, 45, 65-89
16. Papert, S. (1980). Mindstorms. New York, NY: Basic Books.
17. Rich, P. J. & Hodges, C. N. (Eds.) (2017). Emerging Research, Practice and Policy on Computational Thinking. New York, NY: Springer.
18. Richard, G. & Kafai, Y. B. (2016, April). Blind spots in youth DIY programming: Examining diversity in creators, content, and comments within the Scratch online community. In Proceedings of the CHI'16 Conference (pp. 213-227). New York, NY: ACM.
19. Scribner, S. (1984). Literacy in three metaphors. American Journal of Education, 93(1), 6-21.
20. Soloway, E. & Spohrer, J. (Eds) (1989). Studying the Novice Programmer. Norwood, Ablex.
21. Vakil, S. (2018). Equity in computer science education. Harvard Educational Review, 88(1), 26-53.
22. Vee, A. (2017). Coding Literacy: How Computer Programming Is Changing Writing. Cambridge, MA: The MIT Press.
23. Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classroom. Journal of Science Education and Technology, 25, 127-147.
24. Wing, J. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.