Alignment and representation in computer science: an analysis of picture books and graphic novels for K-8 students

Rachelle Haroldson & Dave Ballard

To cite this article: Rachelle Haroldson & Dave Ballard (2021) Alignment and representation in computer science: an analysis of picture books and graphic novels for K-8 students, Computer Science Education, 31:1, 4-29, DOI: 10.1080/08993408.2020.1779520

To link to this article: https://doi.org/10.1080/08993408.2020.1779520

Published online: 18 Jun 2020.

COMPUTER SCIENCE EDUCATION
2021, VOL. 31, NO. 1, 4–29
https://doi.org/10.1080/08993408.2020.1779520

Alignment and representation in computer science: an analysis of picture books and graphic novels for K-8 students

Rachelle Haroldson and Dave Ballard

Teacher Education, University of Wisconsin-River Falls, River Falls, WI

ABSTRACT
Background and Content: Many children's books related to computer science have been published in the last five years, creating opportunities to integrate these texts into the classroom.
Objective: Determine where the children's books support an inclusive computing culture by representing people with diverse intersectional identities engaging in the computer science practices from the K-12 Computer Science Framework.
Method: We analyzed 45 picture books and graphic novels published between 2015 and 2019 targeted at K-8 students. We compiled evidence of the seven computer science practices and representation of diverse characters.
Findings: Three or four practices appear in 64% of the books. The characters that engage in computing in the books are 56% female and 38% people of color. The books offer few examples of nonfictional people of color engaged in computer science, with no representation of adult males of color.

ARTICLE HISTORY
Received 25 September 2019
Accepted 4 June 2020

KEYWORDS
Children's literature; computational thinking; literacy; diversity; culturally responsive computing; K-12 Computer Science Framework

Introduction

With the United States focusing on computer science through movements like CS For All (Smith, 2016), proposed bills to have computer science count for a science, math, or language credit "Senate Bill Counts Coding for Language Requirement.", 2018), and the publication of the computer science standards (Computer Science Teachers Association [CSTA], 2017), we are seeing more schools and districts implementing policies to develop or expand computer science education (2019 Code.org Advocacy Coalition, 2015; McIntyre, 2016; State of Computer Science Education, 2019). This implementation involves teacher professional development, statewide policies and resources, and collaboration with national organizations like Code.org. Teachers need curricular materials to meet these growing demands and guidance about which texts, specifically children's literature, effectively address computer science concepts and engage underrepresented groups through culturally responsive computing.

Modern campaigns in the United States like We Need Diverse Books (WNDB) advocate " … essential changes in the publishing industry to produce and promote literature that reflects and honors the lives of all young people" (We Need Diverse Books, n.d.) and employ the tenets of critical race theory (Mabbott, 2017). We extend their efforts into the

CONTACT Rachelle Haroldson rachelle.haroldson@uwrf.edu

field of computer science, discussing the available children's literature that can be used to promote an inclusive computing culture. Our primary research questions ask what computer science practices are demonstrated in the books and the racial, age, and gender identities of the people engaging in those computer science practices. Using culturally responsive computing as a framework (Scott et al. (2015), we perform a content analysis on 45 children's picture books and graphic novels published in the United States. We track the representation of computer science concepts (as outlined by the K-12 Computer Science Framework) and the intersectional identities of people doing computer science. Following the analysis is a discussion of where and how the books succeed and fall short in creating an inclusive computing classroom for a wide range of youth. We conclude with implications for creating a culturally responsive computing culture.

Review of the literature

Children's literature in the classroom can directly support the teaching of various concepts, including computer science. Other fields have analyzed children's picture books and the ways they portray content and cultural groups. Kelly (2018) examined children's trade books for images of scientists, discussion of the nature of science, and scientific disciplines. She noted an overemphasis on life science topics and a limited description of the nature of science. She found that most books continued to primarily picture white males as scientists, with some progress showing female scientists and no progress showing scientists of different races. She argued that culturally relevant instruction in science supports students' learning, especially when students are provided with resources that project inclusive ideas about who can be a scientist. Computer science educators have echoed this sentiment, recommending more diverse role models, ethnocomputing, culturally situated design tools, and culturally responsive computing to reduce the bias and disparities within computer science (CSTeachingTips (Producer), n.d.; EngageCSEdu, n.d.b.; Morales-Chicas et al., 2019).

Researchers in the field of children's literature recognize the need for culturally relevant books that honor and represent the lived experiences of students across many aspects of identity (Braden & Rodriguez, 2016; Durand & Jimeénez-Garcia, 2018; Hickey & Hopenwasser, 2013; Kelly, 2018). Children need stories they can relate to and that help them understand a common humanity. The contemporary movement for inclusive children's literature, We Need Diverse Books, aims to put books into the hands of all children where they can see themselves in the story (We Need Diverse Books, n.d.). In WNDB's call for more books by diverse authors and about diverse protagonists they build on the work of historical advocates like librarian Charlemae Rollins and author-educator Nancy Larrick (Larrick, 1965; Mabbott, 2017). Rudine Sims Bishop (1990), another advocate, used the term *mirrors* for books written to reflect the lives of diverse students. She advocated for mirror books both for children from under-represented groups (for the purpose of self-affirmation) and children from dominant social groups (to view and meet people unlike themselves). Boyd et al. (2015) argue that students need a richer diet of books depicting " … diverse lifestyles, opportunities, beliefs, choices and worldviews in addition to books that focus on the status quo" (p. 380) because only then will students be able to move beyond the "single story"

(Adichie, 2009) of books being only for and about white, middle-class, heterosexual characters.

To create an inclusive computing culture, computer science educators must bring in resources with images and narratives representing diverse computer scientists, with emphasis on females and people of color (EngageCSEdu, n.d.b.). If we want students to see themselves as the kinds of people who do computer science, then they need to have access to resources that serve as mirrors and reflect back their intersectional selves (Scott et al., 2015).

Massey (2015) argues, "In the hands of educators, picture books serve a much greater function than aesthetic reading; they can be a vehicle for the construction of knowledge and for solidifying concepts in a learning environment for older students" (p. 45). Using picture books with both younger and older students activates visual thinking and helps make abstract concepts concrete, especially in areas like science (Massey, 2015). Using and developing abstractions in computer science in itself is abstract. Students are asked to identify patterns and common features to create generalizations as well as evaluate existing abstractions (K–12 Computer Science Framework, 2016).

According to the Common Core State Standards for English Language Arts and Literacy in History/Social Studies, Science and Technical Subjects (National Governors Association Center for Best Practices, Council of Chief State School Officers, 2010), students need to be able to think critically, use reasoning and evidence collection, and attentively read complex pieces of text. The ELA/literacy standards are meant to be used in conjunction with science and technical subjects to support literacy in those areas for college and career readiness. While computer science is not called out specifically, the technical subject standards related to following a multistep procedure precisely, discerning the meaning of symbols, key terms, and domain-specific terms and their use within a technical context; taking information in words from a text into a visual representation or model, comparing and contrasting information, analyzing relationships among concepts in a text, and synthesizing material from various sources certainly support the computer science standards and practices (Computer Science Teachers Association [CSTA], 2017; K–12 Computer Science Framework, 2016; National Governors Association Center for Best Practices, Council of Chief State School Officers, 2010).

Theoretical framework

This analysis of computer science picture books and graphic novels is guided by the culturally responsive computing framework (Morales-Chicas et al., 2019). Culturally responsive computing draws from the work in culturally responsive teaching and culturally relevant pedagogy, taking the ideas of asset building, reflection, and connection and incorporating them into key tenets specific to technology education (Scott et al., 2015). Both culturally responsive teaching and culturally relevant pedagogy promote academic achievement, foster empowering students, acknowledge the assets students bring, and combat deficit thinking (Gay, 2010; G. Ladson-Billings, 1994). Culturally responsive teaching focuses on effective teaching methods that bring in the lived experiences and frames of reference of ethnically diverse students (Gay, 2002). Culturally relevant pedagogy draws on students' cultural referents and empowers students intellectually, socially, emotionally, and politically (G. Ladson-Billings, 1994). Teachers use the core components of academic achievement, cultural competence, and sociopolitical consciousness to plan the curriculum (G. Ladson-Billings, 2011) and use the students' culture to inform the learning (G. Ladson-Billings, 1995).

To help students move away from the misconception that only white people wearing glasses do computer science (Google & Gallup, 2015), we need to implement culturally responsive education strategies and tools. In other words, in order to effectively reach, teach, empower, and motivate underrepresented groups in computer science education, we need to increase the relevance of computing to their lives and backgrounds (Grover & Pea, 2013; Morales-Chicas et al., 2019; Scott & White, 2013). In their revisit of the theory of culturally responsive computing, Scott et al. (2015) outline five tenets to address issues of inequity and disparity in technology and computer science education. In one tenet, "All students are capable of digital innovation" (p. 420) challenges deficit thinking and outlines the importance of having underrepresented groups exposed to technological tasks, engaged in creation of technology, and held to high expectations in technology programs. This starts in the curricular planning process when teachers are tasked with deconstructing and reconstructing the curriculum and resources they use (G. Ladson-Billings, 2011). It includes the books teachers integrate in their lessons.

In another tenet, Scott et al. (2015) argue for learning that focuses on the intersecting socio-cultural identities of students. Culturally responsive computing draws on *intersectionality* from feminist studies, a more complex and inclusive approach involving the intersection of multiple dimensions of identity (McCall, 2005). Technological learning and innovation must involve and honor the intersectional self and offer opportunities for students to identify their multiple selves (Scott et al., 2015; Scott & White, 2013).

Study context

To date, no analysis has been done on children's literature in the field of computer science. Given the importance of providing an inclusive computing culture to teach and empower traditionally marginalized groups, we need to understand how and if published texts provide rich and relevant information while reflecting back and integrating the identities of a wide variety of students. The study builds on analysis of science trade books (Brunner & Abd-El-Khalick, 2017; Kelly, 2018) around the diversity of scientists, language used, science disciplines represented, and the nature of science; we now evaluate these concepts within the framework of computer science education. The goal of the study is to examine the representation of computer science as outlined by the K-12 Computer Science Framework (K–12 Computer Science Framework, 2016) and the diversity of people doing computer science. The following research questions guide this study.

- What specific computer science practices do the people in the picture books and graphic novels engage in?
- What are the gender and racial identities of the people doing computer science in the picture books and graphic novels?

Definition

"Computational thinking" refers to a cognitive approach aimed at solving problems systematically, as described by practices 3–6 outlined in the K–12 Computer Science Framework (2016). For a complete list of the practices with definitions, see Table 2. The problem is treated as a set of requirements that can be addressed through well-defined processes (practice 3: recognizing and defining computational problems). Then, specific computational abstractions and processes are identified or created that will aid in solving the problem (practice 4: developing and using abstractions). These processes are assembled into a well-defined unit where the processes interrelate in precisely defined ways (practice 5: creating computational artifacts). Finally, the computational artifact is evaluated and improved (practice 6: testing and refining computational artifacts). This process does not necessarily generate code or circuitry; any systematic approach to a problem in this way can be called computational thinking.

Methods

In this study, we conducted a content analysis of 45 children's computer science picture books and graphic novels published between 2015 and mid-2019. According to researchers in the field of children's literature, content analysis examines *what the text is about* (Galda et al., 2000). Content analysis allows for the systematic quantifiable analysis of texts and visuals using samples classified into distinct categories (Bell, 2001; Saldaña, 2011). Bell (2001) notes that content analysis requires precise research questions, clearly defined variables, and reliable coding. Content analysis yields meaningful evidence from a given set of texts and shows what is given priority. In the sections that follow we discuss the criteria for selecting the texts we analyzed and the coding process.

Criteria for selecting books

Picture books and graphic novels published in the United States from 2015 to mid-2019 were selected for this analysis. We wanted the analysis to be current and guided by the national CS for All initiative, which began in 2015 and officially launched in January 2016.

The books were selected from two searchable databases, NoveList K-8 and WorldCat. Both of these databases are accessible to educators and provide information about the target audience (age and grade range), genre, format, and publisher. NoveList K-8, a database for children's literature, offered searches for fiction texts and WorldCat, a global library catalog, offered searches for non-fiction text and biographies. The Library of Congress was used to build a list of standardized search terms. These subject headings included: "computer science", "computer programming", "computer scientists", and "computer programmers." In NoveList K-8, searching for "computer science" produced 101 titles, "computer programming" produced 43 titles, and "computer scientists" produced 16 titles. WorldCat produced 478 titles for "computer science" and 606 titles for "computer programming" under the content designator "non-fiction." WorldCat produced 52 titles for "computer scientists," 55 titles for "computer science," and 45 titles for "computer programmers" under the content designator "biography."

To address our research questions, we examined books with a narrative component, meaning they told a story through a character or biographical figure, thus both fiction and non-fiction texts were included. We narrowed the selection to print books written in English for the K-8 grade band, the ages when picture books and graphic novels are predominantly used in the classroom. We did not review books written for early childhood or for high school. While we recognize that there are many graphic novels available and read by students in grades 9–12, we found that these texts were significantly longer and denser and would require their own, separate analysis. Texts identified as purely informational, like reference books or textbooks, were excluded because they did not align with the character-focused goals of the study. We did not include books that promoted a specific programming language (i.e., Ruby, Python) or curriculum (i.e., Scratch), nor did we look at books that were "how-to" texts or activity books. After applying our criteria and accounting for overlap, 45 picture books and graphic novels for students in grades K-8 comprised the final list (see Table 1).

Data collection and analysis

Each of the authors read every book twice. The first reading was to become familiar with the story and characters. During the second reading, we recorded the gender and racial identities of the people in the books engaging in computer science or computational thinking. We also identified which computer science practices from the K–12 Computer Science Framework (2016) characters engaged in according to the set of criteria we developed. See Table 2 for a practice-by-practice description and detailed criteria. Each time we saw evidence of a practice in the books, we recorded a description and the page number (or a description of the scene in the book for unpaged books). After these initial readings and analyses, we came together to establish interrater reliability. During these discussions, we often conducted a third reading as we compared evidence for the practices and identities of the characters. In the event of a disagreement around including a practice, we looked to the book for specific language description and compared that to the criteria we developed (see Table 2) and the description of the practice within the K–12 Computer Science Framework (2016). This allowed us to come to agreement regarding each practice.

The identification of gender and racial identities of biographical figures was straightforward. Gender identity for any non-biographical characters was clearly represented through visual context and confirmed in the use of binary pronouns in the text. Determining race proved to be challenging, as often the text did not clearly identify the non-biographical characters' racial identity. We agree with scholars that race is a social construction (Spring, 2004). However, while using race oversimplifies the diversity of human experiences, social constructs like race and gender " … shape how we are organized as a society in the United States" (Botelho & Rudman, 2009, p. 133). For the purpose of this study we used race and gender as a way to classify the images and visuals to address how texts honor intersectional identities, as called for in culturally relevant computing. Characters from a variety of backgrounds were depicted in the books, but due to the limited sample size, ambiguity of many characters' exact racial identities, and the particular perception in the United States that computer programmers are white males, we classified the characters only as "white" or "person of color."

Table 1. Computer science picture books and graphic novels.

Bodden, V. (2017). *Programming pioneer Ada Lovelace*. Minneapolis, MN: Lerner Publications.
Boone, M. (2019). *Ada Lovelace*. North Mankato, MN: Capstone Press.
Borgert-Spaniol, M. (2018). *Grace Hopper: Advancing computer science*. Minneapolis, MN: Abdo Publishing.
Bowen, L. A. & Gennari, J. (2019a). *Power coders: Huey's GUI*. New York, NY: PowerKids Press.
Bowen, L. A. & Gennari, J. (2019b). *Power coders: The missing programmer*. New York, NY: PowerKids Press.
Brown, T. L., Dunn, D. L., & Beck, C. (2019). *Instructions not included: How a team of women coded the future*. New York, NY: Disney • Hyperion.
Di Piazza, D. (2018a). *Google cybersecurity expert Parisa Tabriz*. Minneapolis, MN: Lerner Publications.
Di Piazza, D. (2018b). *Space engineer and scientist Margaret Hamilton*. Minneapolis, MN: Lerner Publications.
Funk, J. & Palacios, S. (2018). *How to code a sandcastle*. New York, NY: Viking.
Hayes, Amy (2017). *Ada Lovelace: First computer programmer*. New York, NY: PowerKids Press.
Karanja, C. & Whitehouse, B. (2019a). *Adi sorts with variables*. North Mankato, MN: Picture Window Books.
Karanja, C. & Whitehouse, B. (2019b). *Adi's perfect patterns and loop*s. North Mankato, MN: Picture Window Books.
Karanja, C. & Whitehouse, B. (2019 c). *Gabi's fabulous functions*. North Mankato, MN: Picture Window Books.
Karanja, C. & Whitehouse, B. (2019d). *Gabi's if/then garden*. North Mankato, MN: Picture Window Books.
Labrecque, E. (2017). *Ada Lovelace and computer algorithms*. Ann Arbor, MI: Cherry Lake Publishing.
Lecocq, M., Archambault, N., von Innerebner, J., & Dengo, R. (2018). *Rox's secret code*. Brooklyn, NY: POW!.
Lew, K. (2018). *Ada Lovelace: Mathematician and first programmer*. New York, NY: Britannica Educational Publishing.
Liukas, L. (2015). *Hello Ruby: Adventures in coding*. New York, NY: Feiwel and Friends.
Liukas, L. (2017). *Hello Ruby: Journey inside the computer*. New York, NY: Feiwel and Friends.
Liukas, L. (2018). *Hello Ruby: Expedition to the Internet*. New York, NY: Feiwel and Friends.
Miller, S. M., Hoena, B., & Brown, A. (2019). *A coding mission (Adventures in Makerspace)*. North Mankato, MN: Stone Arch Books.
McKay, C. R. & Gennari, J. (2019). *Power coders: The chatbot mystery*. New York, NY: PowerKids Press.
Nagelhout, R. (2017). *Alan Turing: Master of cracking codes*. New York, NY: PowerKids Press.
Niver, H. M. (2017). *Tim Berners-Lee: Inventor of the World Wide Web*. New York, NY: PowerKids Press.
Niver, H. M. (2019). *Grace Hopper: Computer scientist and Navy admiral*. New York, NY: Enslow Publishing.
Pelleschi, A. (2017). *Mathematician and computer scientist Grace Hopper*. Minneapolis, MN: Lerner Publications.
Reed, J. (2017). *Computer scientist Jean Bartik*. Minneapolis, MN: Lerner Publications.
Robbins, D. & Knisley, L. (2017). *Margaret and the Moon: How Margaret Hamilton saved the first lunar landing*. New York, NY: Alfred A. Knopf.
Robinson, F. (2016). *Ada's ideas: The story of Ada Lovelace, the world's first computer programmer*. New York, NY: Abrams Books for Young Readers.
Sanchez Vegara, M. I. & Yamamoto, Z. (2018). *Ada Lovelace (Little people, big dreams)*. Minneapolis, MN: Lincoln Children's Books.
Schwartz, H. E. (2018). *Code-breaker and mathematician Alan Turing*. Minneapolis, MN: Lerner Publications.
Singh, K. & Konak, I. (2018). *Ara the star engineer*. Vancouver, Canada: Page Two Books.
Stanley, D. & Hartland, J. (2016). *Ada Lovelace, poet of science: The first computer programmer*. New York, NY: Simon & Schuster Books for Young Readers.
Stone, T. L. & Priceman, M. (2018). *Who says women can't be computer programmers?: The story of Ada Lovelace*. New York, NY: Christy Ottaviano Books: Henry Holt and Company.
Vink, A. & Gennari, J. (2019a). *Power coders: A peculiar sequence of events*. New York, NY: PowerKids Press.
Vink, A. & Gennari, J. (2019b). *Power coders: Day of the gamer*. New York, NY: PowerKids Press.
Vink, A. & Gennari, J. (2019 c). *Power coders: The simulated friend*. New York, NY: PowerKids Press.

(*Continued*)

Table 1. (Continued).

Wallmark, L. & Chu, A. (2015). *Ada Byron Lovelace and the thinking machine*. Berkeley, CA: Creston Books.
Wallmark, L. & Wu, K. (2017). *Grace Hopper: Queen of computer code*. New York, NY: Sterling Children's Books.
Yang, G. L. & Holmes, M. (2015). *Secret Coders*. New York, NY: First Second.
Yang, G. L. & Holmes, M. (2016). *Secret Coders: Paths & Portals*. New York, NY: First Second.
Yang, G. L. & Holmes, M. (2017a). *Secret Coders: Secrets & Sequences*. New York, NY: First Second.
Yang, G. L. & Holmes, M. (2017b). *Secret Coders: Robots & Repeats*. New York, NY: First Second.
Yang, G. L. & Holmes, M. (2018a). *Secret Coders: Potions & Parameters*. New York, NY: First Second.
Yang, G. L. & Holmes, M. (2018b). *Secret Coders: Monsters & Modules*. New York, NY: First Second.

Table 2. Computer science practice descriptions and inclusion criteria.

Practice 1: Fostering an Inclusive Computing Culture	
Developing an inclusive and diverse computing culture where people from a variety of ethnicities, abilities, and genders contribute to the field. Examining and including the viewpoints and perspectives of others when designing computational artifacts and recognizing bias that comes up in the design process.	
Practice 1 Includes	**Practice 1 Does Not Include**
● Characters invited, encouraged, or created an inclusive culture ● Characters recognized and addressed inequity or injustice	● Characters overcame obstacles to succeed in computing ● Narrative only described injustice or inequity
Practice 2: Collaborating Around Computing	
Collaborating with others in pairs or teams to create complex artifacts. Working with others from different backgrounds and with different perspectives.	
Practice 2 Includes	**Practice 2 Does Not Include**
● One character described a computational process while another implemented it ● Characters split up work to design, develop, refine, or communicate about computational artifacts	● A character taught other characters about computing ● Other characters looked on while one character did all of the computational thinking ● A character shared computational artifacts with others
Practice 3: Recognizing and Defining Computational Problems	
The first step in computational thinking, where characters plan their approach. Recognizing problems that can be solved computationally, then breaking those problems into subproblems to make solving the problem manageable.	
Practice 3 Includes	**Practice 3 Does Not Include**
● Characters evaluated whether or not solving a problem using computation is the most efficient solution using a cost-benefit analysis	● Characters recognized the potential of computers or a computational process but did not name specific applications
Practice 4: Developing and Using Abstractions	
The second step of computational thinking, where characters develop solutions to subproblems. Using generalized models to structure and control a computational process or to store, manipulate, and present data.	
Practice 4 Includes	**Practice 4 Does Not Include**
● Characters described, then tried existing abstractions/models ● Characters developed their own abstraction to solve a problem ● Characters used loops or control structures, data structures, binary and other numeric encoding systems, etc.	● Characters solved a problem without explaining/demonstrating the specific models, abstractions, or tools used
Practice 5: Creating Computational Artifacts	
The third step of computational thinking, assembling the processes developed to solve subproblems into a single well-defined unit. Creating a program, procedure, or device out of smaller steps/components.	
Practice 5 Includes	**Practice 5 Does Not Include**
● Characters created a program, app, procedure, or computing device to solve computational problems	● Characters used unspecified or presumed computational processes to create products that are not themselves computational processes (e.g., artwork, text, physical items)
Practice 6: Testing and Refining Computational Artifacts	
The fourth step of computational thinking, improving a computational artifact through an iterative process. Addressing and correcting errors and inefficiencies through systematic testing and/or code review. Evaluation of the computational artifact for performance, reliability, usability, and accessibility.	
Practice 6 Includes	**Practice 6 Does Not Include**
● Characters tried different approaches until they found a suitable solution ● Characters created a new artifact to replace an old one in an attempt to solve the problem in a better way	● Characters explained their reasoning or decisions while they developed their artifact, without testing alternatives
Practice 7: Communicating about Computing	
Sharing computational thinking or artifacts with the broader community. Teaching, presenting, or publishing a computational process or artifact with appropriate regard for intellectual property rights, ownership, and sharing.	
Practice 7 Includes	**Practice 7 Does Not Include**
● Characters shared their computational artifact or taught about computing to a broad audience ● Characters engaged their community explicitly about computing	● Characters had casual conversations about computing ● Characters collaborated by discussing computing while they developed a computational artifact (practice 2)

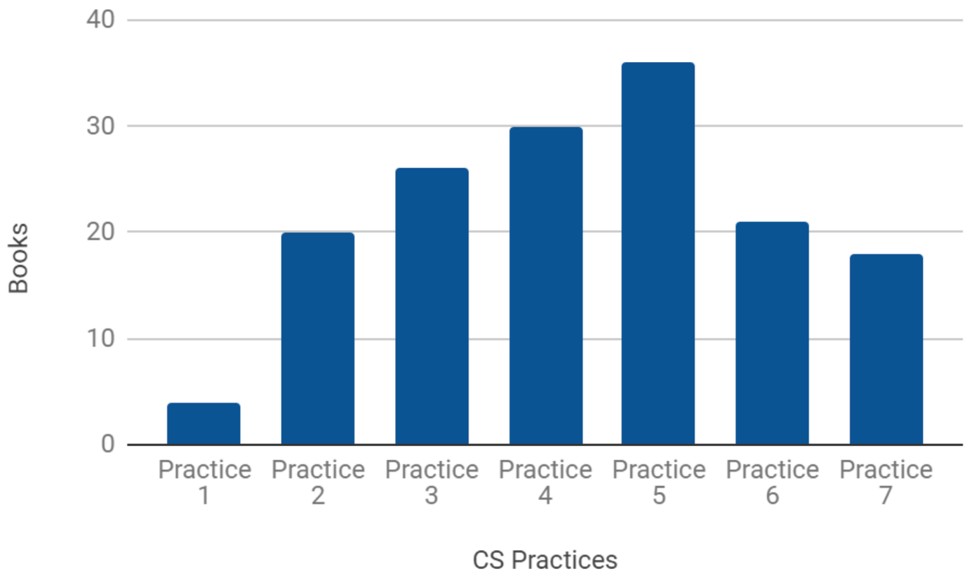

Figure 1. This graph shows the number of books in which each practice appears.

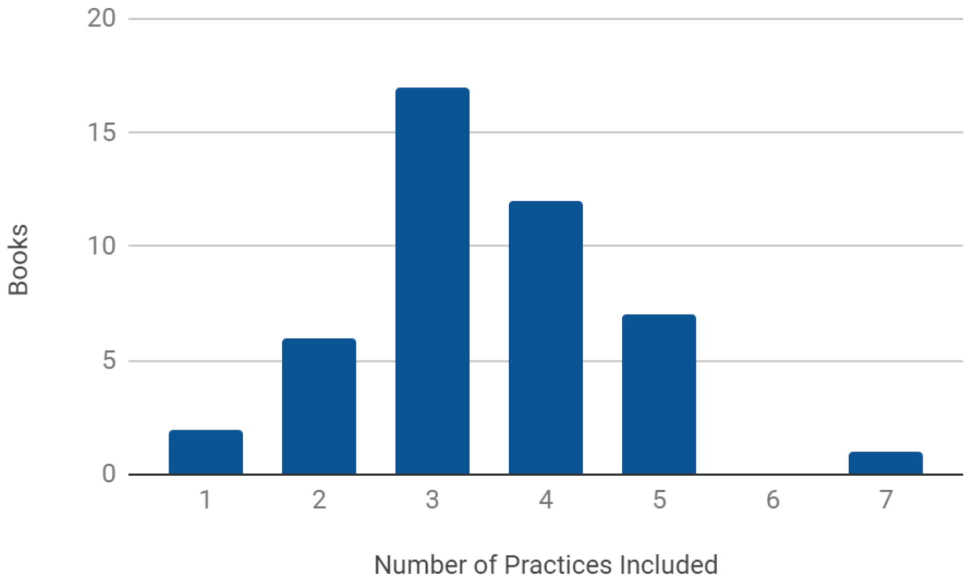

Figure 2. This graph shows the number of books in which N (horizontal axis) practices were included.

To address our first research question, we counted both how many books each computer science practice appeared in and how many distinct practices appeared in each book. After we had gathered this data, we were able to determine which practices were most common and in what type of books a given practice was mostly likely to be found. We were also able to determine how many practices a typical book includes.

To address our second research question, as we tracked characters who engaged in computer science practices, we added them to an overall pool of characters. This allowed us to track not only which and how many books represented characters with various identities, but also to note how many of the total pool of characters represented a given intersectional identity. We tracked general character traits in a binary fashion (the books did not include characters for whom these identity traits were apparently non-binary). We also tracked specific, intersectional identities to find the proportion of the total character pool each identity made up.

Findings

Engaging in the computer science practices

Figure 1 shows the number of books with evidence for each individual practice and the distribution. The books most frequently address practice 3 (addressed in 58% of books), practice 4 (67%), and practice 5 (80%) and practice 1 appears with the least frequency, in only 9% of the books. Practice 2 appeared in 44% of books, practice 6 in 47%, and practice 7 in 40%. Figure 2 shows the distribution of the number of computer science practices. The data shows that 70% of the books reference three or four practices. Only 4% had only one practice and only 2% covered all seven practices.

Practice 1: fostering an inclusive computing culture

Practice 1 shows up the least of all of the practices, with representation in only four books. We considered practice 1 to be present only in books that showed characters actively working toward a more inclusive computing culture; while many books included a diverse cast of characters or mentioned honors received by characters, these did not meet our criteria. A good example of the practice can be found in *Computer Scientist Jean Bartik* (Reed, 2017). Bartik actively fought for women in technology and gave presentations to women, both at university and in industry. She wanted to encourage women to enter and stay in the computing workplace. In *Google Cybersecurity Expert Parisa Tabriz* (Di Piazza, 2018a), Tabriz also encourages women and girls to get involved with technology and computer security through kids' conventions like r00tz Asylum. After creating the World Wide Web, Tim Berners-Lee in *Tim Berners-Lee: Inventor of the World Wide Web* (Niver, 2017) shared it widely and freely with the public. He founded the World Wide Web Foundation, which works " … to keep the web open to everyone and to make sure it helps humanity" (p. 28).

Practice 2: collaborating around computing

Practice 2 appears in 20 books. In *Hello Ruby: Expedition to the Internet* (Liukas, 2018), Ruby, along with her friends Django and Julia, work together to build their Internet out of snow. They all have different ideas about what the Internet is and yet find ways to come together and collaborate for the final product:

> "It was made of small pieces," Julia starts.
> " … all joined together," Ruby continues. "It wouldn't have been the same without all our friends," she adds.
> "We make a great team," Django says with a smile. (Liukas, 2018, unpaged)

In *Secret Coders: Robots and Repeats*, the characters Hopper, Eni, and Josh work together to develop a program to infiltrate the villain's office by tracing a code-shape onto the ground. To trace the shape, they use nested loops. In a classic collaborative "pair programming" approach, Josh acts as the driver at the keyboard, entering the code, while navigators Eni and Hopper describe the code for the inner and outer loops, respectively (Yang & Holmes, 2017b). *Space Engineer and Scientist Margaret Hamilton* (Di Piazza, 2018b) describes how Hamilton worked on a team writing software for SAGE RADAR systems at MIT. Hamilton then led another team to develop and test software for the Apollo Guidance Computer system used by the manned Apollo 11 mission to the moon. *Ara the Star Engineer* (Singh, 2018) offers multiple strong examples of this practice: Ara engages in collaboration with the women she meets on her journey (all real Google engineering leaders) to figure out the number of stars in the sky. The engineers join Ara to help her troubleshoot, brainstorm the algorithm, and develop her code in a pair programming arrangement.

Practice 3: recognizing and defining computational problems

Practice 3 has a high frequency, with references in 26 different books. The main characters, Hopper, Eni, and Josh, in *Secret Coders: Potions and Parameters* (Yang & Holmes, 2018a), identify problems to solve with computational thinking, from deciphering a secret binary message to stopping a truck full of toxic Green Pop with programmable turtle robots. Together they figure out the criteria and constraints involved in a computational approach. In *How to Code a Sandcastle* (Funk & Palacios, 2018), Pearl explicitly calls out the skills involved in this practice, "But a coder takes one big problem and breaks it into smaller ones. If I give Pascal enough instructions that he does know, we'll build this castle in no time!" (Funk & Palacios, 2018, unpaged). Alan Turing works on the problem of breaking the Enigma code in England during World War II. He also imagined a machine capable of solving the "decision problem," the question of whether logic and reasoning could be reduced to pure mathematics, and this led to the development of his theoretical Turing machine in *Alan Turing: Master of Cracking Codes* (Nagelhout, 2017). On the other side of the Atlantic Ocean, Grace Hopper and her team wrote code for the computer Mark 1 to perform calculations so gunners and rockets would hit their intended targets as well as locate and remove mines threatening ships during World War II in *Grace Hopper: Advancing Computer Science* (Borgert-Spaniol, 2018).

Practice 4: developing and using abstractions

Evidence of practice 4 surfaces in 30 of the books. Eni, Hopper, and Josh develop and use lots of abstractions in *Secret Coders: secrets & Sequences* (Yang & Holmes, 2017a). In their after-school session with Professor Bee, they apply parameters, then use if-else statements with Dr. One-Zero. They apply these concepts later to trick Cuddles the robot cat. They continue to apply their use of abstractions, using loops and generalizing discrete, stand-alone sections of code for reuse in *Secret Coders: Potions & Parameters* (Yang & Holmes, 2018a).

Adi and Gabi apply their knowledge of abstraction using functions in a more creative way in the kitchen as they make breakfast. In *Gabi's Fabulous Functions* (Karanja &

Whitehouse, 2019 c) they take the ingredients from the input and using their "function" box, create the output, in this case a yogurt parfait.

> "Let's show your dad our parfait function factory!" Adi says.
> "OK, I'll be the computer," Gabi says. She stands behind the box so that her dad can't see what she's doing. "Input, please" she says to Adi.
>
> Adi hands her the yogurt, berries, and granola. Behind the box, Gabi quickly mixes the ingredients into a fancy glass to make a parfait. Then she sets the finished parfait next to the output sign. (Karanja & Whitehouse, 2019, unpaged)

This is a strong example of young learners using abstraction or repeated pattern (in this case constructing a parfait).

Lovelace used an existing technology she observed in the Jacquard loom to create a powerful and versatile instruction set for computer programming in *Who Says Women Can't be Computer Programmers?: The Story of Ada Lovelace* (Stone & Priceman, 2018). *Mathematician and Computer Scientist Grace Hopper* (Pelleschi, 2017) discusses how Hopper developed subroutines:

> Hopper noticed that the programmers were writing the same code over and over again. This caused them to redo work that someone else had already done. Hopper suggested they write their code one time and put it in a place where everyone else could find it. In notebooks and on loose sheets of paper, the programmers created a library of commonly used code. Hopper called these chunks of code subroutines. (Pelleschi, 2017, pp.15-16)

Eventually Hopper directed her team to put the subroutines in the computer, so the computer could store them for the programmers to retrieve.

The team of Betty Snyder, Jean Jennings, and Kay McNulty developed sort-merge, stored program, and reducing and reusing memory, respectively, as they worked on writing code for the ENIAC computer in their own World War II efforts in *Instructions Not Included: How a Team of Women Coded the Future* (Brown, Dunn, & Beck, 2019). In *Power Coders: Day of the Gamer* (Vink & Gennari, 2019b), the characters are developing an educational game and decide to have the program respond to users in a certain way when they answer a question correctly. The characters discuss and then implement a conditional structure, describing the abstraction while they demonstrate its use.

Practice 5: creating computational artifacts

Practice 5 is the most prevalent of all of the practices, found in 36 (80%) of books in the analysis. *Grace Hopper: Queen of Computer Code* (Wallmark & Wu, 2017) identifies the program Hopper developed:

> Grace invented a program that let people use words to tell the computer what to do. Her program, named FLOW-MATIC, included simple English commands like MULTIPLY. FLOW-MATIC translated MULTIPLY and the other commands into instructions the computer could understand. This was much easier than programming pages of "1"s and "0"s. With the help of Grace's program, she and her coworkers were able to write code more quickly and with fewer errors. (Wallmark & Wu, 2017, unpaged)

The FLOW-MATIC program was remarkable because it would become one of the first compilers (a program used to turn a programming language into machine language), a clear example of an artifact developed for a practical purpose.

In *Code-Breaker and Mathematician Alan Turing* (Schwartz, 2018), Turing develops his theoretical Turing machine (still a foundation of computer science), and later an automatic computing engine and software. The Coders in Yang and Holmes' (2018b) *Secret Coders: Monsters & Modules* also develop a series of artifacts including: a 20-gon algorithm, code for drawing circles to trick the warden in Flatland, code to create a net to catch balloons full of toxic Green Pop, and code to create and move a hero figure to combat the green monster.

Karanja and Whitehouse's (2019b) main characters Adi and Gabi design a code using their train cars in *Adi's Perfect Patterns and Loops*. The girls write the task on each train car, a nod to block-based programming modeled in a concrete way for very young audiences (K-3). Then the train moves around the track following the instructions to create a loop, parallel to loops in computer programs. *Ada Lovelace and Computer Algorithms* (Labrecque, 2017) references Lovelace's use of instructions and tools like looping to create the first known computer program. In *A Coding Mission* (Miller, Hoena, & Brown, 2019) Codie helps her classmates find their way out of the Labyrinth (and away from the Minotaur chasing them) using the random mouse algorithm, the wall-follower algorithm, and the pledge algorithm. Once they get safely back to the Makerspace, Codie leads them in writing the specific code for the pledge algorithm.

Practice 6: testing and refining computational artifacts

Evidence and examples of practice 6 were identified in 21 books. *Gabi's If/Then Garden* (Karanja & Whitehouse, 2019d) provides a clear reference to debugging. In the story, Gabi, acting as the computer, does not perform the expected action when a condition is met. Adi, acting as the programmer, uses the process of elimination to determine what prevented the "computer" from performing as intended. A similar debugging process happens in *Rox's Secret Code* (Lecocq, Archambault, von Innerebner, & Dengo, 2018). When her robot, Chorebot, starts to trample the city, Rox uses the code from another robot, Mischief Bot, to revise and reverse Chorebot's actions. In *Margaret and the Moon: How Margaret Hamilton Saved the First Lunar Landing* (Robbins & Knisley, 2017), Hamilton engages in this process: "With Apollo 11, NASA would finally try to put people on the moon. Had Margaret thought of everything that could go wrong with a lunar landing? She checked her code again to make sure." (Robbins & Knisley, 2017, unpaged). Hamilton's meticulously tested code did work, and her careful refinement saved the Apollo 11 mission from a potential crisis when technical issues arose during the mission. The Power Coders, Tommy, Grace, Naya, and Peter, team up to figure out the location of their missing guest speaker. In *Power Coders: The Missing Programmer* (Bowen & Gennari, 2019b) they work through the program's code to figure out the bug so they can determine where to find coding legend Sam North. Using a flowchart, they determine where the program is missing a line of code.

Practice 7: communicating about computing

Practice 7 is found in 18 of the books in the analysis but was concentrated particularly in biographical non-fiction texts. Of the texts containing examples of characters engaging in this practice, 44% were books about Ada Lovelace that discussed her translation and additions to a paper (see Appendix for a complete list). One example is *Programming Pioneer: Ada Lovelace*, in which Lovelace translates into English Luigi

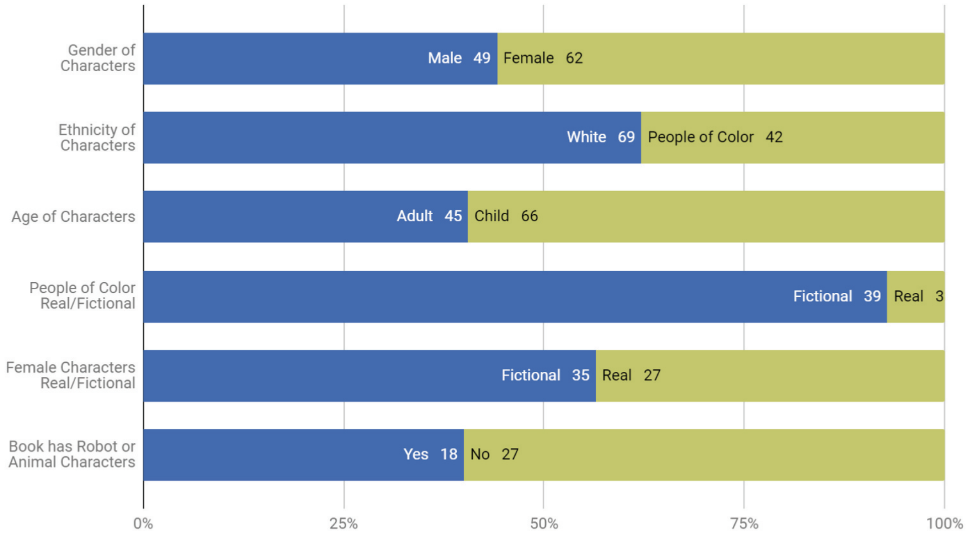

Figure 3. This chart shows six measures of diversity and inclusion in the books reviewed, expressed as the percentage of the books which match each measure.

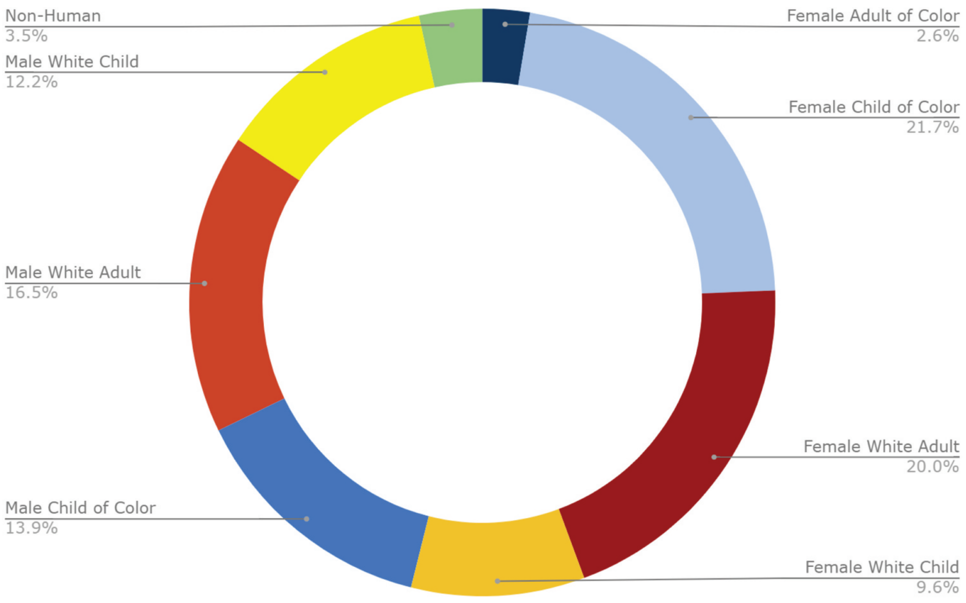

Figure 4. This chart shows the intersectional identities of the characters as percentages of the total pool of characters who engaged in computer science practices in the books.

Menabrea's paper discussing Charles Babbage's analytical engine, communicating with Babbage and adding extensive notes longer than the original paper itself. Her translation was published and well-received and was read and referenced many decades later during the development of electronic computers (Bodden, 2017). Another nonfiction book with a good example of practice 7 was *Grace Hopper: Advancing Computer Science*

(Borgert-Spaniol, 2018), in which Hopper gave lectures and actively encouraged others to get involved in computer science. However, practice 7 did appear in some fiction texts; in *Power Coders: The Chatbot Mystery* (McKay & Gennari, 2019), the characters present first about chatbots, then their own chatbot program to a panel of judges and peers.

For a book-by-book list of which practices appear in which books, see the Appendix.

Gender and racial identities of people doing computer science

Across the 45 books, there are 66 child characters and 45 adult characters doing computer science. Of these characters, 56% are female and 44% are male. Based on the pictures and illustrations, 62% of the characters are inferred to be white and 38% of the characters are inferred to be people of color, including Latinx and African/African American. Among the 62 females portrayed, 56% are fictional and 44% are non-fictional. Among the 42 characters inferred to be people of color, 93% are fictional and 7% are historical or living computer scientists. The books show animals or robots doing computer science 40% of the time. Figure 3 displays these statistics in detail.

Of the 66 children illustrated across the books, 22% appear to be female children of color, 14% appear to be male children of color, 12% appear to be male white children, and 10% appear to be female white children. For the 45 adults featured across the books, 20% are white females, 17% are white males, and 3% are females of color. There were no books depicting adult males of color. See Figure 4 for a representation of the intersectional identities of characters that appear in the books.

Discussion

The purpose of the study was to determine 1) the specific computer science practices people were doing in the books and 2) the identities of the people portrayed doing computer science. Indeed, the findings confirm that the characters and people within these books are doing the computational thinking practices associated with computer science. Specifically, they are engaging in activities like identifying problems that can be solved computationally, using abstractions, developing computational products, and tweaking those products in an iterative process, all of which comprise the process of computational thinking. The books provide readers with a wide variety of information related to computer science. Those books more biographical in nature tended to have descriptions about people creating programs or other artifacts. For example, the books about Ada Lovelace discussed the paper and notes (i.e., the first computer program) she wrote for the Analytical Engine. The books about Grace Hopper described the computer compiler (FLOW-MATIC) and program (COBOL) she created. The fictional picture books tended to be more specific about the types of abstractions they were using and employed specific vocabulary related to loops, sequences, functions, parameters, Boolean expressions, etc. All of the books addressed at least one practice from the K-12 CS Framework (2016) and each book included at least one computational thinking practice.

The findings also confirm that characters collaborate and communicate. The *Secret Coders* series and *Ara the Star Engineer* had the most frequent examples of characters collaborating around computing. Tasks to solve a problem or develop an artifact were delegated, then everyone would contribute based on their strengths. In some cases, the characters would teach each other to help other characters develop, making the team even stronger. The majority of communication about computing in the books involved presenting about artifacts or teaching a group of peers, as shown in the *Power Coders* series, or sharing through publication or formal lectures, a common example in the books highlighting a biographical figure.

While some of the books can be used to promote an inclusive culture in the classroom because of the diverse characters represented, the books themselves rarely specifically addressed this practice. The few that did involve actual computer scientists (e.g., Parisa Tabriz, Tim Berners-Lee, Jean Bartik) and how through their outreach efforts they work or worked to include more people in computer science.

Our findings matched previous studies regarding the representation of diverse characters. About 27% of the characters in the books were inferred to be children of color, very close to the 23% of books identified by the Cooperative Children's Book Center (CCBC), School of Education at the University of Wisconsin-Madison in their 2018 analysis of children's books (Huyck & Park Dahlen, 2019). More books in CCBC's analysis depicted animals (27%) than children of color, parallel to our own finding that more books featured robots and animals (37%) than children of color. While we recognize that animals or robots are used as engaging literary devices for younger audiences drawn to anthropomorphized characters (Kole, 2020), they should not overshadow human characters with a range of intersectional identities.

The findings shed light on the representation of the intersectional aspects of identity. While there appears to be more parity with respect to gender in the books, female children of color and white adult women have the most representation. Fourteen of the total 19 biographical books that highlighted women feature one of two computer science pioneers, Ada Lovelace or Grace Hopper, demonstrating minimal diversity within the adult white females represented. Male children were depicted as white or of color with about the same percentage, and both were less common than female children of color. Female white children were portrayed the least across the children. Given that there have been so many initiatives targeted at young girls (i.e., Girls Who Code, Black Girls Code), one would expect more balance across racial identities like that for male children.

Female adults of color had the smallest percentage across the groups that appeared, even less than that of robots or animals doing computer science. Male adults of color were not represented in any of the books. Both findings are cause for concern when we think about designing an inclusive classroom and including role models in the classroom to encourage more children from underrepresented groups to engage in computer science, particularly Latinx, African/African American, and American Indians/First Nations students. This is especially problematic at the younger grades when there is such a strong push for literacy. Children need access to books where they see characters like themselves doing computer science.

Certainly, many authors of fictional books, like Karanja's Adi and Gabi series, Yang and Holmes' *Secret Coders* series, and the *Power Coders* series, are intentionally including diverse characters. We applaud their inclusive contributions. At the same time, we want to call attention to the lack of non-fiction books highlighting historical or living female and male people of color. To echo groups like We Need Diverse Books and scholars in children's literature, we need diverse stories to foster inclusion, acceptance, personalization, and representation of complex identities (Durand & Jimenez-Garcia, 2018; Kucirkova, 2016; Leahy & Foley, 2018; We Need Diverse Books, n.d.). We need more books with diverse role models and protagonists in order to respond to culturally responsive computing's call for reflecting and representing students' intersectional selves, so they can engage in class, build confidence, and feel included (CSTeachingTips (Producer), n.d.; EngageCSEdu, n.d.a; Scott & White, 2013; Scott et al., 2015). In order to establish an inclusive classroom culture, teachers need access to rich, culturally relevant texts that they can use as tools to teach about diversity (Leahy & Foley, 2018).

Implications for promoting culturally responsive computing

Computer science education researchers recognize the positive impact of culturally responsive computing to engage and empower underrepresented groups in computing (Ashcraft & Eger, 2015; Morales-Chicas et al., 2019; Scott & White, 2013). Culturally responsive computing provides the structure for educators to think about how they will foster an inclusive culture in their classrooms by connecting to the lives and lived experiences of their students. This study focused on the culturally relevant computing tenets around recognizing that all students have the ability to develop technical innovation and intersectionality (Scott et al., 2015). By assuming all students can succeed in computing, we move away from deficit thinking and recognize the funds of knowledge students bring with them (Gonzalez & Moll, 2002). We argue that by providing texts that reflect the identities of different students, teachers start to build that inclusive culture. Students see people that mirror their intersectional selves in the texts (tenet 3) doing complex technological tasks, oftentimes being strongly encouraged by teachers or adult educational mentors (tenet 1). For example, in the *Secret Coders* series Eni, Hopper, and Josh all reflect different socio-cultural identities, including different races, genders, and family structures. They are encouraged throughout the series by Mr. Bee, a former teacher of the Bee School turned custodian, who guides them as they work against conspiring forces. *Power Coders* also has a group of students from a range of backgrounds. They work together with constant encouragement from their teacher Ms. Jones to navigate complex tasks like debugging a program and writing a chatbot app. While Rox in *Rox's Secret Code* does not have an adult mentor, her friend Amar encourages her to fix the problem in the code of Chorebot, then later in the story they push on gender expression stereotypes when Amar wears Rox's tutu. These are but a few examples where youth with identities along intersecting sociocultural lines are doing tasks involving computational thinking: determining the problem, using abstractions, creating artifacts, and refining those artifacts.

Literature and informational texts are key aspects of English Language Arts standards. Documents like the Next Generation Science Standards [NGSS] (NGSS Lead States, 2013) are aligned to the ELA Common Core State Standards (National Governors Association Center for Best Practices, Council of Chief State School Officers, 2010) to support teachers in making their classes explicitly inter-disciplinary. Designing lessons that promote literacy and computer science at the same time can address multiple needs in classrooms where time is limited. For example, The *STEM Trailblazer BIOS* series offers insight into the contributions of figures, past and present, while calling out the academic language of computer science. Having books around to engage learners in computer science topics also provides an unplugged and accessible approach. The *Hello Ruby* books provide such additional unplugged activities in the back of the book for teachers to do with their students to reinforce the different concepts covered. Books can be used to hook students into a topic or to elaborate on an idea or develop understanding of an abstract concept. The *Secret Coders* series provides concrete examples of abstractions (e.g., loops, conditionals, parameters) and computational artifacts. Overall, the books in this analysis teach the computational thinking practices in accessible ways, and teachers are encouraged to use them in their classes, whether those are specific computer science classes or other subjects weaving in computer science.

Providing texts where students of color can relate to biographical figures they read about continues to be challenging. With minimal diversity across the nonfiction texts, teachers will struggle to achieve a culturally responsive computing classroom. Certainly, there are computer scientists of color as well as computer science advocates of color out in the world. A few of those role models for African/African American, Latinx, and American Indians/First Nations students include, but are not limited to: Kimberly Bryant (founder of Black Girls Code), John Henry Thompson (inventor of the scripting language Lingo), Mark Dean (co-creator of the IBM personal computer in 1981), Clarence "Skip" Ellis (computer scientist), Roy L. Clay, Sr. (computer programmer), Evelyn Boyd Granville (IBM computer programmer), Stephanie Castillo (developer of Latina Girls Code), Andrea Delgado-Olson (founder of Native American Women in Computing), and Nicole Archambault (creator of La Vie en Code) (Reichard, 2017; Streeter, 2017; Women of Silicon Valley, 2017). By offering materials with the stories and accomplishments of these historical and modern computer scientists, teachers can help students find connection to people with a shared intersectional identity.

Limitations

As white, cisgendered, English-speaking researchers, we acknowledge that we see through our lens and therefore biases. The inferences we made about characters' identities were likely influenced by our own backgrounds and perceptions. We recognize that others may read these books and characters differently based on their own identities and lived experiences, and we hope that they will add their own perspectives to our findings.

This study only addressed computational thinking when it was illustrated specifically in one of the computer science practices (K–12 Computer Science Framework, 2016). However, these books contained many rich examples of computational thinking that did not necessarily involve the creation of computational artifacts, the particular focus of the computer science practices. This may be an area to explore further in future analysis.

This paper is limited to the analysis and discussion of the content of the selected books. While we have included data (see Appendix) we expect may be useful to teachers designing their own instruction (including, for example, in which books specific practices and representation can be found), we have not included lessons plans or curricula, that include these books. Such practical classroom application of our findings is outside the scope of this study but may be the subject of future research.

Conclusion

To foster an authentic/inclusive computing culture, we need to look to culturally responsive computing. This means focusing on having high expectations of all students, giving them access to technological tasks, and honoring their socio-cultural identities. The analysis of the children's books and graphic novels in this study provides a starting point for educators in selecting materials to use and incorporate into their classes, both to promote literacy and computer science practices. More efforts are still needed to develop and distribute texts highlighting real computer scientists of color and their contributions to reflect the diverse range of students' intersectional identities.

Acknowledgments

This work was supported by the National Science Foundation under Award #1660135; the National Science Foundation under Award #1439768; the University of Wisconsin-River Falls College of Education and Professional Studies Research, Creative and Scholarly Activity Award; and the University of Wisconsin-River Falls Faculty Research Grant.

Special thanks to Dr. Jaime Ballard, Dr. Barb Billington, Amanda Moeller, Dr. Earl Blodgett, Dr. Jacob Hendricks, and Mary Taris who provided feedback on earlier drafts.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the National Science Foundation [1439768]; National Science Foundation [1660135].

Notes on contributors

Rachelle Haroldson is a clinical associate professor at the University of Wisconsin-River Falls and Master Teacher in the STEM*teach* program. Her research interests are culturally relevant pedagogy, science education, computer science education, and formative assessment. She works with graduate students simultaneously pursuing their master's and teaching licenses in science, math, and/or computer science.

Dave Ballard is a physics and computer science teacher in Richfield, MN. He is a former software developer and is currently working on his Master of Science in Education at the University of Wisconsin-River Falls. He read many of the books discussed with his two young children.

References

Adichie, C. N. (2009). *The danger of a single story [Video file]*. https://www.ted.com/talks/chimamanda_ngozi_adichie_the_danger_of_a_single_story?language=en

Ashcraft, C., & Eger, E. (2015). *How can you engage a diverse range of girls in technology?* https://www.ncwit.org/sites/default/files/resources/_howcanyouengageadiverserangeofgirlsintechnology.pdf

Bell, P. (2001). Content analysis of visual images. In T. V. Leeuwen & C. Jewitt (Eds.), *Handbook of visual analysis* (pp. 10–34). SAGE Publications.

Bishop, R. S. (1990). Mirrors, windows, and sliding glass doors. *Perspectives: Choosing and Using Books for the Classroom*, *6*(3), 1–2.

Botelho, M. J., & Rudman, M. K. (2009). *Critical multicultural analysis of children's literature: Mirrors, windows, and doors*. Routledge.

Boyd, F. B., Causey, L. L., & Galda, L. (2015). Culturally diverse literature: Enriching variety in an era of common core state standards. *The Reading Teacher*, *68*(5), 378–387. https://doi.org/10.1002/trtr.1326

Braden, E. G., & Rodriguez, S. C. (2016). Beyond mirrors and windows: A critical content analysis of Latinx children's books. *Journal of Language & Literacy Education*, *12*(2), 56–83. http://jolle.coe.uga.edu/wp-content/uploads/2016/11/56-83_JoLLE_2016_202_Braden__Rodriguez.pdf

Brunner, J. L., & Abd-El-Khalick, F. (2017). Representations of nature of science in U.S. elementary science trade books. In C. V. McDonald & F. Abd-El-Khalick (Eds.), *Representations of nature of science in school science textbooks: A global perspective* (pp. 135–151). Routledge.

Code.org Advocacy Coalition. (2015). *Nine policy ideas to make computer science fundamental to K-12 education*. https://code.org/files/Making_CS_Fundamental.pdf

Computer Science Teachers Association [CSTA] (2017). *CSTA K-12 computer science standards, revised 2017*. http://www.csteachers.org/standards

CSTeachingTips (Producer). (n.d.). *Tips for reducing bias[YouTube video]*. http://csteachingtips.org/tips-for-reducing-bias

Durand, E. S., & Jimeénez-Garcia, M. (2018). Unsettling representations of identities: A critical review of diverse youth literature. *Research on Diversity in Youth Literature*, *1*(1), article 7. https://sophia.stkate.edu/rdyl/vol1/iss1/7

EngageCSEdu. (n.d.a). *Engagement practices: Build student confidence and professional identity*. https://www.engage-csedu.org/engagement/build-student-confidence-professional-identity

EngageCSEdu. (n.d.b). *Engagement practices: Grow inclusive student community*. https://www.engage-csedu.org/engagement/grow-inclusive-student-community

Galda, L., Ash, G. E., & Cullinan, B. E. (2000). Research on children's literature. In M. L. Kamil, P. B. Mosenthal, P. D. Pearson, & R. Barr (Eds.), *Handbook of reading research: Volume III* (pp. 351–381). Erlbaum.

Gay, G. (2002). Preparing for culturally responsive teaching. *Journal of Teacher Education*, *53*(2), 106–116. https://doi.org/10.1177/0022487102053002003

Gay, G. (2010). *culturally responsive teaching* (2nd ed.). Teachers College Press.

Gonzalez, N., & Moll, L. C. (2002). *Cruzando el Puente*: Building bridges to funds of knowledge. *Educational Policy*, *16*(4), 623–641. https://doi.org/10.1177/0895904802016004009

Google & Gallup. (2015). *Images of computer science: Perceptions among students, parents, and educators in the U.S*. https://services.google.com/fh/files/misc/images-of-computer-science-report.pdf

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Hickey, P. J., & Hopenwasser, C. B. (2013). Lingua Anglia: Bridging language and learners: engaging miguel: Culturally relevant YA literature. *The English Journal*, *103*(2), 105–107. https://www.jstor.org/stable/24484202

Huyck, D., & Park Dahlen, S. (2019, June 19). *Diversity in Children's Books 2018. sarahpark.com [web log]. Created in consultation with Edith Campbell, Molly Beth Griffin, K. T. Horning, Debbie Reese, Ebony Elizabeth Thomas, and Madeline Tyner, with statistics compiled by the Cooperative Children's Book Center, School of Education*. University of Wisconsin-Madison. https://readingspark.wordpress.com/2019/06/19/picture-this-diversity-in-childrens-books-2018-infographic/. http://ccbc.education.wisc.edu/books/pcstats.asp

K–12 Computer Science Framework. (2016). http://www.k12cs.org

Kelly, L. B. (2018). An analysis of award-winning science trade books for children: Who are the scientists, and what is science? *Journal of Research in Science Teaching*, *55*(8), 1188–1210. https://doi.org/10.1002/tea.21447

Kole, M. (2020, January 10). *Anthropomorphic animals in children's fiction*. [web log]. https://www.goodstorycompany.com/blog/anthropomorphic-animals

Kucirkova, N. (2016). Personalisation: A theoretical possibility to reinvigorate children's interest in storybook reading and facilitate greater book diversity. *Contemporary Issues in Early Childhood*, *17*(3), 304–316. https://doi.org/10.1177/1463949116660950

Ladson-Billings, G. (1994). *The dreamkeepers: Successful teachers of African American children*. Jossey-Bass.

Ladson-Billings, G. (1995). Toward a theory of culturally relevant pedagogy. *American Education Research Journal*, *32*(3), 465–491. https://doi.org/10.3102/00028312032003465

Ladson-Billings, G. (2011). "Yes, but how do we do it?": Practicing culturally relevant pedagogy. In J. L. Landsman & C. W. Lewis (Eds.), *White teachers/diverse classrooms: Creating inclusive schools, building on students' diversity, and providing true educational equity* (2nd ed., pp. 33–47). Stylus Publishing, LLC.

Larrick, N. (1965, September 11). The all-white world of children's books. *Saturday Review of Literature*, *48*(37), 63–65, 84–85. https://www.unz.com/print/SaturdayRev-1965sep11-00063

Leahy, M. A., & Foley, B. C. (2018). Diversity in children's literature. *World Journal of Educational Research*, *5*(2), 172–183. https://doi.org/10.22158/wjer.v5n2p172

Mabbott, C. (2017). The We Need Diverse Books campaign and critical race theory: Charlemae Rollins and the call for diverse children's books. *Library Trends*, *65*(4), 508–522. https://doi.org/10.1353/lib.2017.0015

Massey, S. R. (2015). The multidimensionality of children's picture books for upper Grades. *English Journal*, *104*(5), 45–58. https://www.jstor.org/stable/24484579

McCall, L. (2005). The complexity of intersectionality. *Signs: Journal of Women in Culture and Society*, *30*(3), 1771–1800. https://doi.org/10.1086/426800

McIntyre, E. (2016). The difficult realities of implementing #CS for All. *Education Digest*, *81*(2), 28–32. https://www.educationdive.com/news/the-difficult-realities-of-implementing-csforall/418127/

Morales-Chicas, J., Castillo, M., Bernal, I., Ramos, P., & Guzman, B. L. (2019). Computing with relevance and purpose: A review of culturally relevant education in computing. *International Journal of Multicultural Education*, *21*(1), 125–154. https://doi.org/10.18251/ijme.v21i1.1745

National Governors Association Center for Best Practices, Council of Chief State School Officers. (2010). *Common core state standards English language arts*.

NGSS Lead States. (2013). *Next generation science standards: For states, by states*. The National Academies Press.

Reichard, R. (2017, September 28). *10 Latinas making their mark in the STEM world*. https://remezcla.com/lists/culture/latinas-stem-science-technology-engineering-mathematics/

Saldaña, J. (2011). *Fundamentals of qualitative research*. Oxford University Press.

Scott, K. A., Sheridan, K. M., & Clark, K. (2015). Culturally responsive computing: A theory revisited. *Learning, Media and Technology*, *40*(4), 412–436. https://doi.org/10.1080/17439884.2014.924966

Scott, K. A., & White, M. A. (2013). COMPUGIRLS' standpoint: Culturally responsive computing and its effect on girls of color. *Urban Education*, *48*(5), 657–681. https://doi.org/10.1177/0042085913491219

"Senate Bill Counts Coding for Language Requirement." (2018, July 12). https://www.languagemagazine.com/2018/07/12/senate-bill-counts-coding-for-language-requirement/

Smith, M. (2016, January 20). *Computer science for all [web log]*. https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all

Spring, J. (2004). *Deculturalization and the struggle for equality* (4th ed.). McGraw-Hill.

State of Computer Science Education. (2019). https://advocacy.code.org/

Streeter, J. (2017, February 1). *7 Black pioneers in computer science [infographic]*. https://blog.newrelic.com/culture/black-history-month-computer-science-infographic/

We Need Diverse Books. (n.d.). https://diversebooks.org/about-wndb/

Women of Silicon Valley. (2017, November 29). *Just 18 really awesome Native folks in STEM*. https://medium.com/women-of-silicon-valley/just-18-awesome-native-folks-in-stem-134211ff14cd

Appendix

Book Title	Author, Illustrator	Year	Suggested Audience by Grade Level	Female Characters	Characters of Color	Child Characters	Practice 1	Practice 2	Practice 3	Practice 4	Practice 5	Practice 6	Practice 7
A Coding Mission (Adventures in Makerspace)	Shannon McClintock Miller, Blake Hoena, Alan Brown	###	3–5	X	X	X			X		X	X	
Ada Byron Lovelace and the Thinking Machine	Laurie Wallmark, April Chu	###	4–12	X		X			X		X	X	
Ada Lovelace	Mary Boone	###	1–3	X							X		X
Ada Lovelace (Little People, Big Dreams)	Maria Isabel Sanchez Vegara, Zafouko Yamamoto	###	4–8	X		X				X			
Ada Lovelace and Computer Algorithms	Ellen Labrecque	###	K-3	X						X	X		X
Ada Lovelace, Poet of Science: The First Computer Programmer	Diane Stanley, Jessie Hartland	###	4–8	X		X			X		X		X
Ada Lovelace: First Computer Programmer	Amy Hayes	###	3–7	X				X	X	X	X		X
Ada Lovelace: Mathematician and First Programmer	Kristi Lew	###	2–4	X						X	X		X
Ada's Ideas: The Story of Ada Lovelace, the World's First Computer Programmer	Fiona Robinson	###	6–9	X					X		X		X
Adi Sorts with Variables	Caroline Karanja, Ben Whitehouse	###	5–8	X	X	X		X	X	X			
Adi's Perfect Patterns and Loops	Caroline Karanja, Ben Whitehouse	###	5–8	X	X	X		X		X	X		
Alan Turing: Master of Cracking Codes	Ryan Nagelhout	###	3–7						X	X	X		X
Ara the Star Engineer	Komal Singh, Ipek Konak	###	K-3	X	X	X		X	X	X	X	X	

(*Continued*)

(Continued).

Book Title	Author, Illustrator	Year	Suggested Audience by Grade Level	Female Characters	Characters of Color	Child Characters	Practice 1	Practice 2	Practice 3	Practice 4	Practice 5	Practice 6	Practice 7
Code-Breaker and Mathematician Alan Turing	Heather E. Schwartz	###	7–11						X	X	X		X
Computer Scientist Jean Bartik	Jennifer Reed	###	7–11	X			X		X		X		X
Gabi's Fabulous Functions	Caroline Karanja, Ben Whitehouse	###	5–8	X	X	X		X	X	X	X		
Gabi's If/Then Garden	Caroline Karanja, Ben Whitehouse	###	5–8	X	X	X				X	X	X	X
Google Cybersecurity Expert Parisa Tabriz	Domenica Di Piazza	###	7–11	X			X	X	X	X	X	X	X
Grace Hopper: Advancing Computer Science	Megan Borgert-Spaniol	###	8–11	X					X	X	X		X
Grace Hopper: Computer Scientist and Navy Admiral	Heather Moore Niver	###	2–5	X							X		
Grace Hopper: Queen of Computer Code	Laurie Wallmark, Katy Wu	###	4–9	X							X	X	
Hello Ruby: Adventures in Coding (1)	Linda Liukas	###	4–8	X		X			X	X		X	
Hello Ruby: Expedition to the Internet (3)	Linda Liukas	###	4–8	X		X		X			X		
Hello Ruby: Journey Inside the Computer (2)	Linda Liukas	###	4–8	X		X				X		X	
How to Code a Sandcastle	Josh Funk, Sarah Palacios	###	4–8	X	X	X			X	X	X	X	
Instructions Not Included: How a Team of Women Coded the Future	Tami Lewis Brown, Debbie Loren Dunn, Chelsea Beck	###	K-3	X				X	X	X	X	X	
Margaret and the Moon: How Margaret Hamilton Saved the First Lunar Landing	Dean Robbins, Lucy Knisley	###	4–8	X					X		X	X	
Mathematician and Computer Scientist Grace Hopper	Andrea Pelleschi	###	7–11	X					X	X	X		X

(*Continued*)

(Continued).

Book Title	Author, Illustrator	Year	Suggested Audience by Grade Level	Female Characters	Characters of Color	Child Characters	Practice 1	Practice 2	Practice 3	Practice 4	Practice 5	Practice 6	Practice 7
Power Coders: A Peculiar Sequence of Events	Amanda Vink, Joel Gennari	###	4–6	X	X	X		X		X		X	
Power Coders: Day of the Gamer	Amanda Vink, Joel Gennari	###	4–6	X	X	X		X		X			
Power Coders: Huey's GUI	L. A. Bowen, Joel Gennari	###	4–6	X	X	X		X			X	X	
Power Coders: The Chatbot Mystery	C. R. McKay, Joel Gennari	###	4–6	X	X	X						X	X
Power Coders: The Missing Programmer	L. A. Bowen, Joel Gennari	###	4–6	X	X	X		X		X		X	X
Power Coders: The Simulated Friend	Amanda Vink, Joel Gennari	###	4–6	X	X	X		X	X	X	X		
Programming Pioneer Ada Lovelace	Valerie Bodden	###	7–11	X					X	X	X		X
Rox's Secret Code	Mara Lecocq, Nathan Archambault, Jessika von Innerebner	###	4–8	X	X	X			X		X	X	
Secret Coders (1)	Gene Luen Yang, Mike Holmes	###	8–12	X	X	X		X		X	X		
Secret Coders (2): Paths & Portals	Gene Luen Yang, Mike Holmes	###	8–12	X	X	X	X	X		X	X	X	
Secret Coders (3): Secrets & Sequences	Gene Luen Yang, Mike Holmes	###	8–12	X	X	X		X	X	X	X	X	
Secret Coders (4): Robots & Repeats	Gene Luen Yang, Mike Holmes	###	8–12	X	X	X		X	X	X	X	X	
Secret Coders (5): Potions & Parameters	Gene Luen Yang, Mike Holmes	###	8–12	X	X	X		X	X	X	X	X	
Secret Coders (6): Monsters & Modules	Gene Luen Yang, Mike Holmes	###	8–12	X	X	X		X		X	X		

(*Continued*)

(Continued).

Book Title	Author, Illustrator	Year	Suggested Audience by Grade Level	Female Characters	Characters of Color	Child Characters	Practice 1	Practice 2	Practice 3	Practice 4	Practice 5	Practice 6	Practice 7
Space Engineer and Scientist Margaret Hamilton	Domenica Di Piazza	###	7–11	X				X			X	X	
Tim Berners-Lee: Inventor of the World Wide Web	Heather Moore Niver	###	K-3				X		X		X		X
Who Says Women Can't be Computer Programmers?: The Story of Ada Lovelace	Tanya Lee Stone, Majorie Priceman	###	6–9	X		X			X	X			X