

Preface

Emerging Research, Practice, and Policy on Computational Thinking

Computational thinking is quickly becoming an essential literacy for modern learners (Wright, Rich, & Leatham, 2012). The International Society for Technology in Education recently defined computational thinking as students “develop[ing] and employ[ing] strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions” (<http://www.iste.org/standards/standards-for-students-2016>). The US Bureau of Labor Statistics projects that the need for computer programmers will be three times greater than the current number of computer science graduates. Perhaps more importantly, many contend that people in a diversity of fields will need to demonstrate the ability to think computationally and to manipulate technology to advance our abilities in different fields (Wing, 2009). Thus, computing is no longer a topic of study just for programmers, but for all pupils.

At the time of this writing, computing has become a compulsory topic in school in over a half dozen different countries, with over a dozen more committing to make it so by 2020 (Balanskat & Englehardt, 2015), including a handful that ask students to learn computing from the earliest levels. With this increasing attention to the need to teach computing at earlier ages comes the responsibility to study and understand effective teaching and learning practices in computing education, especially for non-computer science students. Up to this point, most attention has focused on changing policies and practices. This volume is an attempt to bring together emerging research around computational thinking, from primary education to high school and on through higher education.

We issued a call for chapters and received dozens of submissions, demonstrating that there are many who are studying the emerging practices of teaching computing to new groups. The chosen 25 chapters represent authors from nearly a dozen different countries. Each chapter was first reviewed editorially and then peer-reviewed by other authors. The resulting volume is divided into six different sections: (a) K-12

education, (b) higher education, (c) teacher development, (d) assessment practices in computational thinking, (e) computational thinking tools, and (f) computational thinking policies.

Chapters in *K-12 education* represent the ways in which computing has been implemented in a public school context during the formative years. D’Alba and Huett share the result of working with middle and high school students in an after-school context in which parents played a collaborative role in learning to code. Seneviratne presents the successes and challenges of working specifically with high school girls, a typically underrepresented group in computer science education. Jones-Harris and Chamblee describe the results of working with African-American students in a precalculus course. Delcker and Ifenthaler describe how one German state is implementing a new required computer science course in their upper secondary schools. Finally, Tatar, Harrison, Stewart, Frisina, and Musaeus explore the type of thinking that must first occur in order to prepare middle school students to begin to think computationally and its associated challenges.

Higher education chapters present a look at the ways in which computational thinking is being integrated with university courses to improve instruction across a diverse range of subjects. Musaeus, Tatar, and Rosen explore what computational thinking would look like in medical education and how this might improve students’ analytical abilities. Rambally demonstrates how computational thinking can improve students’ abilities to understand discrete mathematical structures. Quaye and Dasuki propose a method for teaching computational thinking in an introductory programming course in Nigeria using a virtual world context. Kaya and Cagiltay show how focusing on computational thinking in an introductory computing course for non-CS majors can successfully lead to increased understanding of core concepts. Liu, Perera, and Klein report on efforts to teach computational thinking while promoting collaboration, problem-solving, and the sharing of educational resources.

The *teacher development* section focuses on the preparation of teachers who formerly had received no training in computing and how they might successfully teach it to their pupils. The chapters by Hester-Croff and Buss and Gamboa both report on efforts to train teachers to foster and recognize computational thinking patterns in middle and high school students. Both Yadav, Gretter, Good, and Mclean and Sadik, Ottenbreit-Leftwich, and Nadiuzzaman focus on how to prepare preservice teachers during their formative teacher education to teach computational thinking. Toikkanen and Leinonen report on the design and executing of a MOOC with over 1000 teachers as they prepare for the mandatory integration of computing in primary education in Finland.

The section on *assessment practices in computational thinking* attempts to better understand what and how to measure as students engage with computational thinking practices. Mueller, Becket, Hennessey, and Shodiev analyzed Canada core competencies for alignment with computational thinking practices. In so doing, they found several areas where computational thinking might already be assessed in related topics and discuss areas for improvement and integration. Grover proposes a system of assessments that go beyond measuring just cognitive outcomes. She

then demonstrates how this is being applied in a middle school course on foundations of computational thinking.

The *tools* section shares resources for teaching, assessing, and implementing computational thinking. Repenning, Basawapatna, and Escherle present a framework for different tools that scaffold learners through three important stages of computational thinking: problem formation, solution expression, and execution/evaluation. Lawanto, Close, Ames, and Brasiel demonstrate how the Dr. Scratch tool can be used to measure students' computational thinking skills. Similarly, Brasiel et al. outline the development, use, and research of the FUN! Tool, a Python-based framework for measuring minute-by-minute interactions in the Scratch environment.

Policy chapters demonstrate the changing landscape and efforts of different governments and groups as they attempt to implement computational thinking. Pan describes the restructuring of CS0, a required introductory computing course for all non-CS majors in the People's Republic of China, to include computational thinking principles and practices. Similarly, in response to the Korean government's designation to include software education as part of its core curriculum by 2018, Lee presents an exposition of representative projects created from a group of 72 pilot schools nationwide that showcase how computational thinking might be integrated into the curriculum. Ruberg and Owens describe a district-wide approach to implementing computing from kindergarten all the way through grade 12, including both in-school and after-school efforts to integrate computing across the curriculum. Finally, Kafai and Burke, two of the early promoters of computational thinking, end the book with a reconceptualization of computational thinking to the broader notion of computational participation, an idea that promotes the integration of computing into communities of practice that extend beyond simply problem-solving and creating to also including individual and group expression and everyday solutions to life.

Together, these chapters paint a picture of the emerging research and practice surrounding efforts to include the teaching and learning computational thinking in an increasingly computational world. While they are by no means a comprehensive report of all that is occurring, it is our hope that they are representative of the challenges and successes that students, teachers, and policy makers face as computing becomes an essential and required subject of study.

Provo, UT, USA
Statesboro, GA, USA

Peter J. Rich
Charles B. Hodges

References

- Balanskat, A., & Englehardt, K. (2015). *Computing our future. Computing programming and coding. Priorities, school curricula and initiatives across europe*. European Schoolnet.
- Wing, J. M. (2009). Computational thinking. *Journal of Computing Sciences in Colleges*, 24(6), 6–7.
- Wright, G. A., Rich, P., & Leatham, K. R. (2012). How programming fits with technology education curriculum. *Technology and Engineering Teacher*, 71(7), 3.

Emerging Research, Practice, and Policy on
Computational Thinking

Rich, P.; Hodges, C.B. (Eds.)

2017, XXVII, 413 p. 60 illus., 41 illus. in color., Hardcover

ISBN: 978-3-319-52690-4