0 An Important and Timely Field

Sally A. Fincher and Anthony V. Robins

Computing education (CEd) is important, everyone agrees. President Obama committed hundreds of millions of dollars to "Computer Science for All" (White House, 2016); governments have developed curricula and made computing a school subject across the world (see Chapter 18); online providers compete to teach coding (such as Code Academy, code.org, the Hour of Code, Khan Academy and Coursera); and tech giants put money into supporting CEd projects (CISCO supports the BlueJ and Scratch initial programming environments and Google funds substantial professional development programs and has produced a series of CEd research reports – one of which forms the basis for Chapter 3).

With all of the effort and resources going into CEd, it would be comforting to think that we know what we are doing – that the problems of teaching and learning computing topics are well understood, that the solutions are known, and that best practice is widely shared. But this ideal picture is very much a work in progress. We still don't know enough about how students learn computing subjects, what effects different teaching approaches have, or how to equally engage people of all races and genders in the field. CEd research (CEdR) is how we work to understand and improve this. In order to make the most of the resources currently going into CEd efforts around the world, CEdR is an important and timely field.

0.1 Why It's Important

The sudden interest and investment in CEd are no accidents. Computing technology is reshaping the world around us at an ever-increasing pace, changing the way that we work (or don't work), communicate, consume, learn, create, entertain ourselves, and more. At the time of writing, some of the major computing-driven issues being widely debated include: the rise of artificial intelligence applications (automation of work, digital assistants, self-driving vehicles); the use and abuse of social media and personal data; and the advent of disruptive cryptocurrencies.

In this context, there are many rationales as to why CEd and therefore CEdR are important. Blikstein and Moghadam (Chapter 3 of this handbook, summarizing their Google report into the current state of CEd) outline four of

them. *The labor market rationale*: computing skills are explicitly required for an increasingly large number of jobs and will be generally useful for very many more. *The computational thinking rationale*: computational ways of thinking (e.g., algorithms, heuristics, problem-solving skills) are all useful and transferable. In practical terms, the belief that they are good preparation for later specialist topics (e.g., learning to program) has helped to drive the widespread introduction of computational thinking in schools (these and related issues are focuses of Chapters 17–20). *The computational literacy rationale*: general familiarity with programming and other computing skills is sometimes equated with mathematical or textual literacy (Chapters 18 and 19). Some further argue that computational literacy goes beyond computational thinking to enable new types of mental operations, knowledge representations, and modes of expression (see Chapter 3). *The equity of participation rationale*: computing knowledge will be increasingly required for the best jobs, for civic participation, and even for understanding the functioning of the society around us. As is readily apparent, current participation in CEd has a long way to go to achieve anything like equity of participation in terms of race or gender (see Chapters 16 and 24).

0.2 Why It's Timely

In 2004, when *Computer Science Education Research* (Fincher & Petre, 2004) was written, CEd researchers were essentially all academics teaching in university computer science departments. Almost universally they had strong disciplinary computing backgrounds and their loci of research were their interests, which meant (almost universally) tertiary-level CEd. In the intervening 15 years, times have changed. Researchers in schools of education are becoming interested in the field, there are many agencies funding research, and there are increasing numbers of specialist academics with PhDs in CEd. The locus of research has expanded to include adult "returners" and children in K–12 education (which adds the complexity of cognitive development to the mix). And the subject itself has expanded beyond its academic disciplinary construction in university computing departments, bursting through the classroom walls into everyday life and "computational thinking."

One of the ways in which these changes are reflected is in our decision, as editors, to use the inclusive phrases "computing" education (CEd) and "computing" education research (CEdR), which the reader will find in widespread use in this handbook, instead of the previously common "computer science" phrasing.

As interest in the subject explodes, and as teaching and learning of computing happen in ever more diverse ways in ever more diverse environments, so CEdR must keep pace. This handbook is a contribution to the widening discourse – in it, we capture what is already known, look out to what is known in other fields, and examine what we might be moving toward as computing technologies continue to evolve and our knowledge of CEd develops.

0.3 How This Book Is Organized

All of the chapters are new and have been written explicitly for this handbook (Chapters 3 and 23 draw on previous sources). However, we wanted this to be more than just a collection of writings from a collection of interesting authors. We solicited chapters in three **parts**. The purpose of the **Background** part is to briefly orient the reader within the field of CEdR, its history, and its current status. The substantive Foundations and Topics parts each have a specific purpose and a different set of shared **themes** running "vertically" through their chapters.

The **Foundations** part serves a "textbook" function. It is intended to set our field in context and to be a practical guide for new researchers. In 2005, the Association for Computing Machinery's Special Interest Group in Computer Science Education (SIGCSE) held a panel discussion on challenges to CEdR. Panelists commented on the "isolation" of our discipline, that "Too much of the research in computing education ignores the hundreds of years of education, cognitive science, and learning sciences research that have gone before us." They reflected on "the challenge of diversifying the research methods we employ" and on the need to understand our methods and seek rigor (Almstrum et al., 2005). Looking back on these challenges more than ten years later, although there has been progress, we think it is fair to say that every one of them remains relevant today. This handbook works to address those concerns, to provide an overview of CEdR work and methods, and to show how they fit with other intellectual traditions.

Some of the chapters in this part are broadly concerned with "methods," both quantitative and qualitative (Chapters 4–7). Although a reader might expect to find similar chapters – certainly similarly titled chapters – in many books, our authors have closely contextualized these within CEdR work. Note that there are two extensive chapters on statistical methods. In our opinion (frequently reinforced while reviewing in various contexts), the need for an improved level of statistical rigor is a particular priority within our field.

CEdR (like education research more broadly) borrows techniques (and terminology and methods) from other disciplines in a "trading zone" activity. The idea of an intellectual "trading zone" was first proposed by Peter Galison in his work on physics:

> I intend the term "trading zone" to be taken seriously, as a social, material, and intellectual mortar binding together the disunified traditions of experimenting, theorizing, and instrument building [in subcultures of physics]. Anthropologists are familiar with different cultures encountering one another through trade, even when the significance of the objects traded – and of the trade itself – may be utterly different for the two sides.
> (Galison, 1997)

It is one of the goals of this handbook to situate CEdR with related fields, our "intellectual trading partners," from whom we have much to learn, and to whom we have much to offer: Chapters 8–11 in the Foundations part seek to

do just that. They each articulate expertise from key partner disciplines and explore their boundaries and our common edges. Over time, of course, the nature of the trading zones and their borders shift. These chapters focus on historical CEd trading zones where researchers trained as computer scientists encountered unfamiliar epistemologies and methods. As computing becomes a school subject and becomes an integrated part of other disciplines (such as bioinformatics), new and important trading partners will emerge.

The authors of chapters in the Foundations part were asked to addresses three particular themes: what can we learn **empirically**, **methodologically**, and **theoretically** from their distinctive, separate foundational fields. Themes are signaled in different ways in various chapters, sometimes as section headings, sometimes as bold keywords in relevant places in the text.

The **Topics** part contains chapters that explore the "state of the art." We have chosen topics that are of current and abiding interest in CEdR to illustrate the kinds of problems that we are trying to address and why they matter to us. Very often in this part, chapters draw on a considerable body of existing work, which should help orient new researchers. The themes of this part were **motivational context** (why we care about this issue), **implications for practice**, and **open questions/**suggestions for future research. Once again, these themes will be reflected in various ways in the chapters.

CEdR is essentially applied research, and there is no point in doing this kind of research if you are not interested in affecting practice and making CEd better in some way, perhaps more effective or more equitable. So "implications for practice" was chosen as a theme for this part, as a reminder both to researchers and practitioners that our work is meant to be useful.

The chapters in this part are grouped into four subsections. *Systematic Issues* (Chapters 12–16) are the "bread and butter" areas that continually offer questions and often attract the interest of new researchers. Chapters in the *New Milieux* section (Chapters 17–20) consider more recent issues that have arisen with the spread of computing beyond the "traditional" university setting, situated in a formal classroom within a department of computing. The *Systems and Software Technology* section (Chapters 21–23) recognizes a disciplinary advantage that CEd researchers have, in that we can build computational tools both in support of CEd and to provide new lenses for CEdR. The *Teacher and Student Knowledge* section (Chapters 26–29) investigates issues concerned with the production and acquisition of computing knowledge.

0.3.1 Case Studies

We conclude the Topics part with two case studies. They are written from a different viewpoint and serve a different purpose to other chapters. Chapter 30, *A Case Study of Peer Instruction*, covers the life cycle of an intervention, demonstrating how results from CEdR may be directly applied to practice. Chapter 31, *A Case Study of Qualitative Methods*, details the progress of a paper from inception to publication. This is an uncommon view, and a valuable one. Many CEd

researchers come from computing backgrounds, with a grounding in analytic knowledge. Seeing how qualitative research "plays out" in a study, and in presentation, broadens understanding.

0.4 How This Book Was Written

Although talked about in a desultory fashion for a period of time, a definite point marked the genesis of this handbook. A fortuitous period of study leave following Dagsthul seminar 16072 (Assessing Learning in Introductory Computer Science) allowed the prospective editors to be in the same place for an extended time.

We met frequently over a period of weeks to write the proposal, mapping out an initial list of chapters and identifying a pool of possible contributors (many of whom we had already talked with at Dagsthul and other community venues, such as the annual International Computing Education Research conference). Once the proposal had been reviewed and accepted, we formally invited contributors, directing them to a Google document indexing the proposal, our provisional chapter list, and other relevant resources. Over the first few weeks, building on feedback from both reviewers and contributors, the chapter list was modified and extended, and prospective authors "voted" for the topics and chapters they were interested in contributing to. From this, a lead author was identified for each chapter and writing teams were organized. An initial deadline for chapter completion was set.

Every chapter was written as a Google document, viewable and editable by all contributors to the handbook. We established the convention that only the authors writing a particular chapter were expected to edit the main text, with other contributors suggesting changes, leaving marginal comments, or adding notes in preface pages specifically designated for that purpose. This totally open model was designed to encourage a high level of peer review and discussion, and in that it was moderately successful. Collectively, the contributors left 2,070 comments on their own and each other's chapters, and many more extended notes in the preface pages. Most chapters benefited significantly from this communal sharing of knowledge.

This was a high-trust model from the outset, and we were alert to a number of infringements that would be possible. We were worried that the model might risk authors making unauthorized changes to another person's text; that there might be public and unresolvable disputes between authors; that text might be "borrowed" and published elsewhere; and, in good open source style, that someone (or some group) might decide to disrupt the project. None of that happened.

There were minor abuses, which are common to all models of collective action (Ostrom, 1990). There was some *free-riding* (where contributors took their share of the benefits, but did not make an equal contribution) and some *rule-breaking* (authors not delivering on time, or at all; authors extending their

authorial invitation to others), but these could have equally occurred in less trustful forms of engagement. A more significant issue – and not obvious to us at the outset – was that some chapters went in different (sometimes very different) directions from what we expected. This was sometimes a good thing, where our initial expectations were poorly informed, but occasionally led to more mixed outcomes.

Our initial deadline came and went (as deadlines will do) with many chapters incomplete. As experienced academics, we had anticipated and allowed for this, and so put the second deadline in place. During this period, we began a systematic review, where every chapter was read and commented on by a group made up of both the editors, other handbook contributors with relevant expertise, and, on occasion, additional subject experts. Each week, a review group met with chapter authors (as time zones allowed) via Skype to discuss reviewers' feedback.

In parallel, a second, independent, and rather unusual review process was taking place. Contributors Shriram Krishnamurthi and Kathi Fisler were teaching a graduate course in CEd at Brown University and suggested their class might read and review the chapters. As graduate students with an interest in the field, they represented one of the key intended audiences for the handbook; this was a great opportunity that we enthusiastically accepted. The class subsequently reviewed almost every chapter (a huge achievement!), and contributors benefited from the additional, external, targeted feedback resulting from this process. At the end of these review processes, authorial teams responded to feedback (some chapters changed markedly in this process) and finalized chapters.

0.5 Accessible Structure

As already mentioned, one of our goals for the Handbook was to present a well-organized body of work, where the structure is made evident and accessible to the reader. This goal drove the organization into parts, and the "vertical themes" running through Parts II and III. We also encouraged authors to make explicit cross-references to other chapters where applicable. This resulted in more than 120 internal references to other chapters to help the reader find related information all over the book (a result that would not have been possible without the open writing process).

As a final effort in this regard, Table 0.1 lists topics that receive substantial attention across several chapters. It is a mix of broad theoretical frameworks (constructivism, cognitivism, behaviorism) and topics specific to our field (learning to program, the notional machine, the McCracken study of novice programmers). We were surprised by some of the entries (Logo commands a lot of attention!). Many other topics are discussed, in varying levels of detail, in more than one chapter. We regard this overlap as a feature, not a bug – from the multiple perspectives of multiple authors, the reader should get a sense of the scope and richness of these topics.

Table 0.1 *Frequently discussed topics and the chapters they occur in.*

Topic	Chapters
Logo (Papert)	1, 3, 8, 17, 19, 20, 22, 27
Constructivism (Vygotsky, Bruner)	1, 8, 9, 10, 11, 15, 24, 29
Teaching methods	1, 8, 10, 12, 15, 24, 27
Learning to program	1, 3, 12, 13, 21, 27
The notional machine	1, 12, 13, 15, 21, 27
Cognitivism	1, 9, 10, 11, 15
Behaviorism	1, 9, 10, 11, 15
Assessment	10, 11, 14, 18, 21
Motivation/efficacy	3, 11, 17, 24, 28
McCracken study	1, 4, 12, 13, 14
Computational thinking	3, 17, 18, 20, 24
Equity/diversity	11, 15, 16, 24

0.6 Looking Back

Our process was "loosely specified." In retrospect, we might have made life easier for ourselves if we had implemented a more formal process. We could have asked authors to submit outlines and drafts of chapters for approval, or we could have asked more people to take on editorial oversight. Such measures would have likely made chapters more consistent and may have led to there being less overlap in some topic areas, with authors "carving up the territory" and claiming the right to reference certain topics exclusively.

What we gained through our process, however, is a collection of strong contributions, with every authorial team working on topics they cared deeply about. Our trade-off is a plurality of views on some topics and idiosyncratic presentation in some chapters.

It is unlikely that this is the first work that has been produced in such an open and collaborative fashion, but it is almost impossible to conceive of such a process being possible in earlier times without the infrastructure of the twenty-first century internet.

0.7 Looking Forward

The handbook is finished, but the work is not. Given the technological, social, and educational changes currently in progress, there is every reason to expect a new wave of interest and participation in CEdR. We hope that the handbook will serve as a useful resource for some time to come as instruction for the novice, a guide for the curious, and a companion for the experienced.

References

Almstrum, V. L., Hazzan, O., Guzdial, M., & Petre, M. (2005). Challenges to computer science education research. In *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education* (pp. 191–192). New York: ACM.

Fincher, S., & Petre, M. (2004). *Computer Science Education Research.* London: RoutledgeFalmer.

Galison, P. (1997). *Image and Logic: A Material Culture of Microphysics*. Chicago, IL: University of Chicago Press.

Ostrom, E. (1990). *Governing the Commons: The Evolution of Institutions for Collective Action*. Cambridge: Cambridge University Press.

White House (2016). Computer Science for All. Retrieved from https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all