8 Learning Sciences for Computing Education

Lauren E. Margulieux, Brian Dorn, and Kristin A. Searle

8.1 Introduction

The learning sciences are an amalgamation of fields that study learning and learning environments. When learning sciences emerged in the 1990s, learning scientists were people who had been training in other disciplinary fields and wanted to apply their skills in multidisciplinary teams to improve learning environments. Some of the fields under the purview of the learning sciences are education, psychology, computer science, educational technology, linguistics, and data analytics. At the time that this book was published, learning scientists were still primarily trained in one of these component disciplines (Yoon & Hmelo-Silver, 2017), though many universities now offer learning-sciences-oriented programs, and some even offer a Masters or PhD in Learning Sciences (Sommerhoff et al., 2018). This shift in training represents a shift in infrastructure for learning sciences work. Many universities have centers for learning sciences in which researchers from various fields can find resources to work together. The professional society, the International Society of the Learning Sciences, was founded in 2002 and organizes annual conferences and supports high-quality publications. Most importantly, learning scientists have developed relationships with places of learning (e.g., K–12 schools, colleges, and museums) so that we can study learning in authentic environments while simultaneously improving the experience of learners right now.

One of the first things that learning scientists discovered when they started working together is that they all have different definitions of learning (Alexander, Schallert, & Reynolds, 2009). Those from cognitive psychology and neuropsychology tend to define learning as a change in the brain – a development of neural architecture and synapses. Those from computer science and educational technology tend to define learning as mastery of a sequence of concepts – a list of rules that build upon each other to allow the learner to understand. Those from education and linguistics tend to define learning as a change in experience – a change in what learners can accomplish and their attitudes about topics or situations, especially as they relate to a sociocultural context. All of these perspectives are considered equal in the learning sciences, and learning scientists intentionally attend to each perspective. For example, a computer scientist might build educational software based on a sequence of concepts that needs to be learned. If the computer scientist was also a learning scientist, they would design the learning experience around how these concepts connect to the

existing cognitive architecture that the learner has and the identity, motivations, and experiences of the learner. Taking a learning sciences perspective would mean that to evaluate the software, we need to not only measure how learners progressed through the sequence, but also what they thought as they progressed and how they applied their knowledge outside of the system.

Among various definitions of learning and among various fields that contribute to the learning sciences, there are a few common tenets that define learning science research. Learning sciences research has the following components (derived from Nathan, Rummel, & Hay, 2016, and Nathan & Sawyer, 2014):

- Design of learning environments and practices based on learning theories (see Section 8.2 on theoretical foundations);
- Application-focused basic research, typically involving mixed methods and design-based research (see Section 8.3 on methodology);
- Authentic practices and settings to test hypotheses and build upon learning theories (see Section 8.3 on methodology);
- An engineering ethos to design and develop new practices and resources (see Section 8.4 on project stages).

These tenets emerged from several central, field-building movements. These movements demonstrated the importance of authenticity and interdisciplinarity (Kolodner, 2004), design-based research (Brown, 1992), computer-supported collaborative learning (Stahl, 2005), technology-enhanced learning environments (Pea, 1994), a broad definition of learning (Yoon & Hmelo-Silver, 2017), and accepting only evidence-based findings to build learning theories (Nathan & Sawyer, 2014). From these tenets and movements, some general research foci that learning scientists share include anchoring learning in prior knowledge, the role of expert knowledge in instruction, learning through social interaction, designing to scaffold levels of understanding, and designing technological supports for knowledge building (Sawyer, 2014).

8.1.1 Learning Sciences and Computing Education: Twins Separated at Birth

The learning sciences and computing education can trace their roots back to a related field: cognitive science. Cognitive science emerged in the 1960s from a combination of fields (see Chapter 9). The two that are relevant here are cognitive psychology and computer science. As these fields grew together, they forged a connection between how humans learn and how machines learn. Herb Simon's group was among the first groups of researchers to model human cognition using computers by making analogies between the human brain and computing processes (Newell & Simon, 1972). By the early 1990s, the field had made great progress in understanding how humans and machines learned, creating learning theories and the foundations of artificial intelligence.

Around this time, a group led by Roger Schank and then Janet Kolodner started to become disillusioned with the epistemology followed in cognitive

science. For example, during the late 1980s and early 1990s, John Anderson developed a cognitive tutor to teach LISP and ultimately proposed the ACT-R theory (seen also in Chapter 9), concluding that human problem-solving boiled down to the mastery and aggregation of problem-solving rules. Anderson (1996) stated that learning to solve problems was simply the sum of its parts, but there were a lot of parts. The first learning scientists, in contrast, argued that this view of learning and the research in cognitive science were too sterile to be applied to authentic learning. Authentic learning includes not only cognitive factors, but also the environment, the instructor, fellow learners, personal attitudes and beliefs, and use of technology (Kolodner, 2004). Therefore, the learning sciences broke from the traditions in cognitive science of highly controlled experiments in lab settings to embrace new practices of application-minded design experiments that are less controlled (and less scientifically rigorous), but more generalizable to authentic learning environments (Hoadley, 2004).

The balance in the learning sciences between the scientific study of learning and the design of environments to support learning is similar to the balance that computing education has embraced since the late 1960s (see Chapter 1). Some computing education researchers from then until now would be considered learning scientists, whether they would describe themselves that way or not. Computer scientists in computing education bring a valuable skill set to the learning sciences – the skills to design and develop learning technologies and environments to support learning. For example, Papert (1980) used Logo, a programming language that allows users to draw with a turtle and makes it easier for the learner to map between the written program and the output, as a technological tool to teach math and problem-solving skills. His work expanded our knowledge of how children learn using an authentic environment that still echoes in how we teach children programming today. Many introductory programming experiences still use drawing with a turtle (e.g., Code.org and PencilCode), and Logo informed the development of Scratch (Maloney et al., 2004; Resnick & Ocko, 1990), the most widely used and researched programming environment for children in primary school (e.g., Kafai & Burke, 2014). The aspect of Papert's work that qualifies it as learning sciences work is that it drew attention to the multitude of epistemological approaches to programming (Turkle & Papert, 1990), emphasizing the importance of social context, including culture, and personal attitudes toward learning a discipline.

Despite overlapping values and some overlapping researchers between computing education and the learning sciences, the two fields have not been as integrated as would be beneficial (Almstrum et al., 2005; Robins, 2015). Computing education researchers tend to focus primarily on developing and evaluating the instruction and tools used in computing education without emphasizing the context of learning (e.g., social factors or personal beliefs) or making connections to more general theories of learning. Of course, there are exceptions (e.g., Ben-David Kolikant & Ben-Ari, 2008; Guzdial & Tew, 2006), but this focus limits the generalizability of computing education research. Without examining the context of learning, educators who want to implement

the instruction or tools in their own learning environments have very little information about how to be successful. In contrast, learning scientists focus more broadly on the design of the learning environment and emphasize development of knowledge about mechanisms and theories of learning, sometimes while contributing only shallowly to discipline-specific education knowledge. These differences play to each group's skills, but both sets of skills are valuable in both fields.

In recent decades, computing education and the learning sciences have made significant steps toward integration. Computing education research has more diligently incorporated learning theory, use of mixed methods, and testing with rigorous statistical analyses (Lishinski et al., 2016; Malmi et al., 2014), which are all common in learning sciences research. In turn, computing education, especially around computational thinking, has become present in learning sciences conferences and journals more regularly (e.g., Margulieux et al., 2016; Orton et al., 2016). Both fields are taking on issues related to equity and bias, particularly concerning learners who are of color, female, from low socioeconomic status families, or with limited access to resources (see Chapter 16 on equity and diversity). This reciprocal relationship benefits both fields and should continue to grow. This chapter provides an introduction to the theories, methods, and practices of the learning sciences, particularly as they relate to computing education, to help those who are unfamiliar with the learning sciences discover the connections between these fields.

8.2 Theoretical Foundations

In this section, we introduce some of the underlying **theoretical** foundations of the learning sciences to discuss what the computing education research community can learn about theory from the learning sciences. The four theories discussed here have long histories of empirical work and represent major components of learning and learning environments. Constructivism addresses how learners cognitively build knowledge; cognitive apprenticeship addresses how instruction scaffolds learners' emerging skills and knowledge; sociocultural theory addresses the social and environmental aspects of learning environments; and expectancy-value theory addresses the role of motivation in learning. Of course, these components do not exist in isolation in the learning environment, and similarly, these theories interact with each other. We will discuss these interactions at the end of this section.

8.2.1 Constructivism

Constructivism is a commonly used theory, both inside and outside of computing education (see Chapters 11 and 15), about how people cognitively acquire knowledge. In essence, constructivism states that people learn best when they construct knowledge for themselves rather than being told explicitly

what to know and how to learn it (Tobias & Duffy, 2009). There is a lot to unpack in that definition. "Construct knowledge for themselves" means that learners are making sense of new information through reasoning and invoking their prior knowledge rather than being told how to interpret and organize new information, as is common in more direct instruction (i.e., instruction in which the instructor explicitly tells the students everything that they need to know). It also means that students are learning concepts and skills through exploration that is guided by an instructor but not prescribed by an instructor, as it would be in more direct instruction. The instructor can still have learning objectives, but there are multiple paths to achieve them. In the definition, "learn best" has several different meanings. It means that constructivist approaches help learners perform better on tasks and tests by increasing their depth of thought and connections to prior knowledge, resulting in better retention and transfer of knowledge (Bruner, 1973; see more about transfer in Chapter 9). It also means improving motivation and emotion by increasing student agency in learning and helping them connect knowledge to their lives (Searle & Kafai, 2015).

The theory of constructivism stems from Piaget's work in cognitive development (as described in Chapter 9). Constructivism became refined, popularized, and applied to instructional strategy in Vygotsky's and Bruner's work starting in the 1960s (as stated in Chapter 10). Unlike the author of Chapter 10, who believes that current work on constructivist pedagogy is nonscientific, the authors of this chapter would call this work scientific even though it does not meet the standards of control found in the hard sciences. We are hardly the first group of people to disagree on this topic. The debate between constructivist and direct instructionist pedagogies reached its peak in the 2000s when Kirschner, Sweller, and Clark (2006) published a paper arguing that all types of minimally guided and unguided learning, which roughly equates to approaches that are fundamentally aligned with constructivism, have not been as effective as direct instruction. Hmelo-Silver, Duncan, and Chinn (2007) and Schmidt, Loyens, van Gog, and Paas (2007) published papers in response to make counterarguments that constructivist learning methods are effective when sufficient guidance is provided by the instructor to the student. In turn, Sweller, Kirschner, and Clark (2007) responded with criticisms of the scientific validity of their evidence, leading to a book edited by Tobias and Duffy (2009) that includes authors from both sides of the debate and allowed them to criticize and respond to the arguments in each other's chapters.

At the center of this debate is a fundamental difference in the definition of learning. The direct instructionists view learning as a change in the brain caused by the storage of new information, and they therefore argue that direct instruction is the most efficient and easiest method for learning. The constructivists view learning as a change in knowledge that is not worth much without a concomitant change in professional skills (e.g., solving authentic problems) and soft skills (e.g., working collaboratively). The latter is much harder to study in true experiments than the former, leading to criticism of scientific rigor by the direct instructionists. Constructivists argued, however, that scientific rigor

is not worth research that is conducted in sterile environments (i.e., labs) that are fundamentally different from the authentic environments (e.g., classrooms) in which the research will be applied (discussed further in Section 8.3.1 on design-based research; Brown, 1992). As with most debates, many researchers and educators fall in the middle, recognizing the contributions of both types of instruction and treating them as two ends of a spectrum. Therefore, instruction can be more direct or more constructive depending on the needs of the learner and what is most appropriate. For example, when novice programmers are first introduced to Java, they will likely learn more efficiently by being told exactly how to write an assignment statement than they will from being asked to come up with their own ideas about how to write an assignment statement. This type of instruction will likely lead to more shallow learning than a less direct approach, but the balance between depth of knowledge and learning efficiency must be considered. If the Java learners were already experienced with Python, though, using less direct instruction and a constructivist activity to scaffold the connection between new Java knowledge to prior Python knowledge will likely help them to learn Java more deeply without significantly impacting efficacy. A core question in the research on constructivism – and learning sciences more generally – is how much guidance is optimal to support learning.

This section only scratches the surface of constructivism as a theory and the instructional strategies that are based upon it, but there are many other places in this book to learn more, especially in the context of computing education.

- Chapter 1 discusses Papert's work on constructionism. Constructionism and constructivism are related, but they are not the same and should not be used interchangeably. Constructionism is based upon constructivism, but it stipulates that the learner should externally construct artifacts to aid the internal construction of knowledge structures.
- Chapter 9 discusses some of the cognitive science theories that relate to constructivism.
- Chapters 24 and 29 discuss the critical social aspects of constructivist pedagogies.

8.2.2 Cognitive Apprenticeship

Early work in the learning sciences drew inspiration from a variety of places, including non-school environments where successful learning has been taking place for centuries. Collins, Brown, and Newman (1989) noted that the vast majority of learning throughout history was structured as a relationship between a master, who is an expert in the domain knowledge and skills to be learned, and an apprentice, who aspires to become a master. The apprentice learns through observation and deliberate practice under the guidance of the master. Tasks are sequenced to gradually increase in complexity in line with the apprentice's emerging skills. Traditional apprenticeships are often associated with trades like carpentry or midwifery, but many academic private tutoring models share similar properties.

There are two primary challenges with apprenticeship-style learning when viewed in a modern school context (Collins & Kapur, 2014; Lave & Wenger, 1991). First, traditional apprenticeships rely on a small student–teacher ratio with one master supervising at most a handful of apprentices at any one time. This level of individualized attention is impractical in a typical classroom setting, and thus it is easy to see how schools evolved direct instruction pedagogies to address the scale of universal education. Secondly, the knowledge and skills acquired in a traditional apprenticeship are narrowly scoped to one specific work domain. The goal of developing generalizable knowledge and skills that is central to modern education seems, at first glance, incompatible with apprentice-style pedagogy (for more information about knowledge transfer, see the transfer section of Chapter 9).

Cognitive apprenticeship was proposed as a means to integrate the successful practices of traditional apprenticeships with the more general knowledge and cognitive skills sought by traditional school settings (Collins, Brown, & Newman, 1989). This approach to orchestrating a learning environment holistically considers four unique components: content, method, sequence, and sociology (Collins & Kapur, 2014). *Content* in this sense is concerned not only with domain knowledge, but also with the heuristic strategies used by experts within the domain to solve problems, the metacognitive control strategies used to monitor one's progress while completing a task, and the more general strategies to learn new things.

A myopic focus on domain knowledge in a learning environment often leaves tacit these strategic components of content, and cognitive apprenticeships seek to avoid this by externalizing both novice and expert strategies explicitly in the learning environment. A hallmark of a cognitive apprenticeship is employing a variety of pedagogical *methods* to synergistically achieve this goal. Expert modeling is used by a teacher to demonstrate a particular task while voicing one's inner thought process for direct observation by the learners. While modeling often precedes the learner's attempt at a task, learning sciences researchers continue to research when and how to model tasks for maximum impact (see Section 8.2.2.1 on productive failure). A variety of coaching techniques are employed by the instructor as students carry out tasks, and the instructor provides scaffolding (Wood, Bruner, & Ross, 1976) artifacts to aid learners along the way. Also central to cognitive apprenticeships is that learners engage in deliberately articulating their knowledge and reasoning, and they have multiple opportunities to reflect on how their approaches compare to those of the experts and other learners. Lastly, learners are encouraged to engage in independent exploration of the problem space. It is important to distinguish this exploration from the type of completely unstructured inquiry criticized by Kirschner, Sweller, and Clark (2006), as described in the previous section. The other strategies must be used carefully in concert to help guide the learning and mitigate demands on a learner's working memory during exploration (Hmelo-Silver, Duncan, & Chinn, 2007).

Consistent with Vygotsky's constructivist Zone of Proximal Development (1978), cognitive apprenticeship learning environments consider the careful

sequencing of learning activities to increase the task complexity and diversity of skill learners develop alongside their growing abilities. Additionally, considering global skills (rather than local skills) first helps orient the learner toward the big-picture tasks to be addressed (Collins & Kapur, 2014). Scaffolding can be provided to abstract away the local skills early on, and learners gradually see more detail as they progress.

Lastly, these learning environments embrace the *socially embedded* and cooperative nature of learning seen in traditional apprenticeship environments. Content is situated in real-world contexts and explored by communities of learners (see Section 8.2.3 of this chapter) while fostering learners' intrinsic motivation (see Chapter 28 of this volume).

At a high level, the elements of cognitive apprenticeship outlined here stake out the multifaceted and holistic research endeavor that is the learning sciences. Each component, like instructor modeling and coaching techniques, learner self-explanation/reflection, and social influences on learning, carries with it a rich body of knowledge and a set of ongoing open research questions to be explored empirically. Indeed, many of these ideas are only just now finding their way into the computer science education researcher literature (see, e.g., Morrison, Decker, & Margulieux, 2016, and Section 8.4 of this chapter), but adopting the systems-level viewpoint that cognitive apprenticeship suggests may both strengthen the theoretical soundness and practical impacts of our work. In the sub-sections to follow, we explore additional theories that underpin some of the practices of cognitive apprenticeship described here.

8.2.2.1 Productive Failure

Productive failure is a learning design that formalizes the process of learning from one's mistakes. Most productive failure research has been carried out in math and science contexts in which many problems have canonical solutions, but a growing body of literature examines how to design for productive failure in less-structured contexts, including computing and engineering tasks, such as debugging. Productive failure has four central mechanisms: activating learners' prior knowledge and experience; drawing attention to critical features of the concept; elaborating on critical features; and integrating critical features into a unified understanding of the targeted concept (http://manukapur.com/productive-failure/). These four interrelated mechanisms are embedded in two phases: a generation phase and a consolidation phase (Kapur, 2015; Kapur & Bielaczyc, 2012). Learners first work in small groups to generate and explore multiple representations and solution methods (RSMs) to an ill-structured problem that is beyond their current problem-solving abilities. Thus, prior knowledge is activated, but failure is encountered because the problem is beyond learners' current problem-solving abilities. In the second phase, the RSMs generated by the learners are compared and contrasted with canonical RSMs and learners consolidate knowledge, integrating the solutions they generated with the targeted

concepts. Ultimately, learners who initially experienced failure when faced with an ill-structured problem are better equipped to solve a well-structured problem as well as subsequent ill-structured problems (Kapur, 2008). Further, the more solutions generated in the first generation and exploration phase, the more knowledge gained, in what Kapur (2015) has called the solution generation effect. Over a series of studies, Kapur and colleagues (2008, 2012, 2014, 2015) have shown similar levels of procedural fluency to direct instruction, in addition to significantly better gains in terms of conceptual knowledge and knowledge transfer (Collins & Kapur, 2014).

8.2.3 Sociocultural Theory

In addition to many theoretical approaches embraced by learning scientists that focus on individual-level cognitive factors, sociocultural theories of learning take a situative perspective on learning. They emphasize learning as an activity that is embedded within social, cultural, and historical context and occurs in interaction with others and with available tools and resources. Sociocultural theory is not a single theory, but rather a group of theories largely growing out of the work by Russian psychologists, including Vygotsky, Luria, and Leont'ev (Sannino, Daniels, & Gutierrez, 2009). Sociocultural theories emphasize the importance of studying learning in real-world contexts rather than laboratories and are uniquely suited to addressing issues of power and equity in learning environments (Esmonde, 2017). Due to space constraints, here we address situated learning (Lave & Wenger, 1991) and activity theory (Engeström, 1987; Greeno & Engeström, 2014).

Exploring professional communities in tailoring, butchering, midwifery, and other trades, Lave and Wenger (1991) argued that learning occurs as an individual moves from being on the edges of a community (legitimate peripheral participation) to more full participation in a community of practice. A community of practice can be understood as a group of people who have in common some form of "practice" such as a type of work or a hobby (Wenger, 1998). As learners move from legitimate peripheral participation to fuller participation in the community, they begin to act like a member of that community, understanding what constitutes community membership, what members of the community do, and how members of the community talk and interact with others, both inside and outside of the community. In other words, learners begin to identify with that community. In such a view, learning *is* identity construction. As learners become full-fledged participants in a community of practice (an academic discipline), they take up new habits and practices associated with that group of people.

In computing education, ideas of legitimate peripheral participation and communities of practice matter in terms of both recognizing the social, cultural, and historical contexts in which computing is situated (Margolis & Fisher, 2002; Margolis et al., 2008) and providing opportunities for learners to take on the identity of a computer scientist (see Chapter 16). For instance, there is a body of

work that examines scaffolding computing education for novices by providing a context for doing computing (Cooper & Cunningham, 2010; Guzdial, 2003, 2010) and fostering opportunities for learners to work together (Porter et al., 2013). Another significant strand of computing education research focuses on addressing the "identity gaps" that exist for women and non-dominant individuals entering into computer science (Tan et al., 2013) and finding ways to mitigate those through more approachable introductions to programming, such as storytelling (Kelleher, Pausch, & Kiesler, 2007), game design (Kafai, 1995), and fashion (Kafai et al., 2014).

Like situated learning, activity theory emphasizes the study of learning at the level of activity systems (e.g., a small group of students working together or an individual learner interacting with tools and materials to make something). Activity systems are composed of interactions between a subject (or group of subjects), an object or overarching goal, and the available tools and resources (Vygotsky, 1978). Further, "tools are created and transformed during the development of the activity itself and carry with them a particular culture – the historical evidence of their development" (Kaptelinin & Nardi, 2006). In this way, tools represent an accumulation of social knowledge and its transmission. For example, if we look at laptop computers or tools for teaching computing to young children, the tools themselves represent an accumulation of knowledge about where, when, and how the tool is used in an activity. Individuals or groups of individuals learn at least some of this knowledge through interaction with the tool.

Engeström's (1987) cultural-historical activity theory (CHAT) further elaborated Vygotsky's model of an activity system to include subject, object, instruments, rules, community, and division of labor. Learning within an activity system is not about individual identity shifts, as is the case in a situated learning perspective, but rather about how the practices of the system as a whole change as a result of a conflict within the system and how that change was accomplished. Further, effective change, according to Engeström, requires an expanded understanding of the object that takes into account both temporal (taking a long view) and socio-spatial elements (Engeström & Sannino, 2010). Activity theory is particularly prevalent in human–computer interaction (Kaptelinin & Nardi, 2006) as a way to understand the role of technology within meaningful activities.

8.2.4 Expectancy-Value Theory

Related to sociocultural theory, much research in learning sciences includes motivational factors, such as learner experience, attitude, values, dispositions, mindsets, and identity. Much more about these factors can be found in Chapter 28. In this section, which is intended to give high-level overviews of theory, we will describe one popular theory of motivation: expectancy-value theory.

Expectancy-value theory is a motivation theory to explain choice, persistence, and performance. Originally developed outside of an education context, it expanded into education in the 1980s with work by Eccles (1983, 1987). It

continues to be a prominent theory for motivation in education today (Wigfield, Tonks, & Klauda, 2009). Expectancy-value theory is primarily applied in K–12 education, but no research suggests that age or developmental stage impacts the predictive value of the theory (Wigfield & Eccles, 2000).

The two components of expectancy-value theory, unsurprisingly, are a learner's expectancy and subjective task value. Expectancy is a learner's belief about whether they can produce a successful outcome for a task. Task value is a learner's subjective assessment of the value of the success or failure of a task's outcome. Task value has four components: attainment value or importance (i.e., importance for self, identity, or community), intrinsic value (i.e., interest and enjoyment), utility value (i.e., usefulness), and cost (i.e., time to achieve, effort to achieve, emotional and physical toll, and trade-off with other valued alternatives; e.g., spending nights at college courses or with family) (Wigfield, 1994).

Expectancy and value interact with each other to predict motivation. When expectancy and value are both high, the learner is highly motivated to successfully complete the task. When expectancy and value are both low, the learner is unmotivated to complete the task. When expectancy and value do not match, they can interact with each other in interesting ways to affect motivation.

- When expectancy is high but task value is low, motivation will generally be low unless task value increases. In some cases, high expectancy can increase the task value, especially for intrinsic value, which is related to enjoyment and interest.
- When task value is high but expectancy is low, motivation will generally be low unless task value is extremely high or expectancy increases. In some cases, low expectancy can decrease the task value by changing subjective evaluation of any of the four components of value (e.g., decrease intrinsic interest or decrease perceived usefulness).

Most interventions related to expectancy-value theory aim to increase motivation, and ultimately achievement, by increasing expectancy or increasing task value, especially by decreasing cost and increasing utility by connecting learning to students' lives (Blackwell, 2002; Blackwell, Trzesniewski, & Dweck, 2007; Hulleman & Harackiewicz, 2009; Wigfield & Eccles, 2000). Though expectancy-value theory seems highly relevant to computing education, it has largely not been used to predict motivation in computing education. More about motivation in computing education can be found in Chapter 28.

8.2.5 Summary

In this section, we have introduced four theories that are part of the foundation of learning sciences work. It is not uncommon to find one or more of these theories discussed in a learning sciences paper, even if the main contribution of the paper does not expand upon them. These theories, and others like them, feed into each other. For example, constructivism is a theory about the nature of knowledge and how learners build knowledge, and it influences how instruction

and learning are implemented in cognitive apprenticeship. Both theories aim to predict the types of scaffolding that help students learn and perform well. Sociocultural theories consider this cognitive development of knowledge and skill as one aspect of the learning environment and examine the impact of social, cultural, and historical context on performance and other critical components of learning, such as a change in identity or experience. Cognitive, social, cultural, and historical components of learning both impact and are impacted by expectancy and value – the parts of expectancy-value theory – and contribute to motivation. Therefore, learning sciences research considers these interwoven connections among theories to design learning environments and the methods used to evaluate them.

8.3 Methodology

In this section, we explore common **methodology** used in the learning sciences to discuss what computing education research can learn about methods from the learning sciences. The methods used in learning sciences research are as diverse as the fields that contribute to it. Much like in computing education research, it is common in conferences and journals to find all kinds of research designs (see Chapter 4) that are analyzed with both qualitative (see Chapter 7) and quantitative (see Chapters 5 and 6) approaches. The most frequently used methods in the learning sciences reflect the origins of the field. This means that researchers pay attention to different definitions of learning and measure both the process of learning (i.e., the experience during and the steps of learning) and the product of learning (i.e., the change in knowledge or experience caused by learning). Research methods and measurements in the learning sciences also make sure to capture the learning environment and its effects. The environment includes features of the immediate surroundings, such as teachers, peers, technology, room, and time, in addition to the more abstract context, such as culture, identity, and family and friend relationships. Not every learning sciences study measures all of these environmental factors, but they do recognize and consider the potential impact that environment might have on the results.

8.3.1 Design-Based Research

A particularly common research method that is used to capture the complexities in learning sciences research is design-based research (DBR). DBR is an evolving research methodology with origins in design experiments (Brown, 1992; Collins, 1992) that is well suited and increasingly used in computing education research (Kelly et al., in press; Shapiro et al., 2017). Using a DBR protocol, a team of researchers will identify a learning problem and potential solution. The learning problem could be a social problem, such as underrepresentation of certain groups in computing; a motivation problem, such as students dropping out of certain types of classes; a knowledge problem, such as students performing

https://doi.org/10.1017/9781108654555.009

poorly on assignments; a cultural problem, such as students (or their teachers) not identifying as someone who could do well in computing; and much more. The team will then identify a group of learners, such as a classroom of students, who are representative of the population that experiences the learning problem. The team will then develop an intervention to address the problem that is specifically designed for the selected group of learners. If the intervention does not work as intended at first, the team will adjust it and iterate as needed. Once the intervention has successfully addressed the learning problem for the first group of learners, the team will identify a new group of learners and adapt and iterate the intervention until it works for the next group. Through several iterations, the team eventually develops an intervention that addresses the learning problem for the entire population of interest.

DBR is based on several ontological viewpoints. First, DBR is rooted in the premise that cognition is inseparable from context, and therefore it is used to design new kinds of learning environments and to research their implementation in the complexity of real-world settings. Second, it is based on the stance that to understand the effect of a variable, that variable must be manipulated while the effects are measured. Therefore, DBR is explicitly interventionist. As a result of these two viewpoints, DBR interventions are studied in design experiments that are positioned to balance the internal validity of experiments, in which researchers attribute differences among groups to the intervention, and the ecological validity of naturalistic settings, in which the manipulated intervention might not be implemented as planned, but the context represents a real learning environment. By working within this balance, DBR is particularly useful for helping researchers to develop theories that explain why something is happening, the conditions under which a particular type of learning or interaction can take place, and the ways in which an individual's mind interacts with the environment and available tools. As a result, DBR sees interventions that change features of environments, activities, or tools as part of the process to be studied.

Studying the process of design experiments in crucial because DBR is both *prospective* and *reflective*. Designs are initially implemented based upon some hypothesized learning trajectory and means of supporting it through a particular design or design feature. However, as the design is implemented, new features of the environment emerge as salient, and both design and implementation may be refined. As a result, iteration is necessary in design to allow designers and researchers to deal with multiple aspects of a learning ecology (Brown, 1992; Collins, 1992). Both design and research take place through cycles of design, implementation, analysis, redesign, re-implementation, and analysis. Therefore, methods and measurements must be able to document all of these phases in order to adequately capture the dynamics of the learning ecology (Cobb et al., 2003).

To capture these dynamic components, DBR uses a collection of methodological approaches that share some common features (Barab & Squire, 2004; Cobb et al., 2003; Design-Based Research Collective, 2003; Edelson, 2002).

DBR has two goals that are intertwined: the design of learning environments and the development of pedagogical theories. This means that theories are often mid-level and populations are typically more narrow than in psychological research. For example, instead of attempting to create a theory-based intervention that would work for all novice programmers, as psychological research would, DBR would focus on ninth- or tenth-grade novice programmers in a particular region or using a particular curriculum. As Cobb et al. (2003) elaborate, "Rather than grand theories of learning that may be difficult to project into particular circumstances, design experiments tend to emphasize an intermediate theoretical scope (DiSessa, 1991) that is located between a narrow account of a specific system (e.g., a particular school district, a particular classroom) and a broad account that does not orient design to particular contingencies" (p. 11). Theories developed through DBR must do real work in the world, facilitating sharing with practitioners and other designers while improving educational outcomes for participants. As Hermes, Bang, and Marin (2012) articulate in thinking through an Ojibwe language revitalization project, "DBR … has the affordance of engaging educational researchers in developing immediate solutions for critical, timely, and practical problems in education" (p. 384).

By focusing on design, DBR positions itself to focus on innovation. Edelson (2002) argues that one of the main benefits of DBR is that it puts researchers in real learning situations with a somewhat open-ended focus on improvement, opening the door to learn unique lessons and developing original interventions. In other words, much DBR demands a break from business as usual in classrooms, schools, and other educational contexts. For this reason, DBR research must have buy-in from everyone who is invested in the educational context. DBR is typically carried out by teams of researchers working in partnership with administrators, teachers, students, parents, and other community members. It also demands active, engaged participation from the team of researchers to refine theories and measurement as the work progresses.

If creating real change within educational contexts in a relatively rapid time period is one of DBR's greatest strengths, it is also one of its greatest weaknesses. The theories and designs generated through DBR are often critiqued as being too formative in nature, the timescale too condensed (Barab, 2014). Further, in spite of its focus on situating learning in context, DBR has been relatively silent about the roles that culture and sociohistorical context play in schooling and design more generally. Ironically, "the lessons involved in DBR often uncover the sociohistoric foundations in which learning, education, and language are deeply entrenched" (Hermes et al., 2012, p. 384). In other words, while DBR has not historically focused on issues of culture and power, these sociohistoric issues are uncovered as a result of DBR.

8.3.2 Lessons to Learn from DBR for Computing Education Research

DBR is very relevant to computing education research, and computing education research is well suited to using DBR (see Section 8.4). Aligned with

the goals of DBR, computing education research often focuses on innovation. With a relatively short history, learning environments in computing are open to changes in instruction and tools. Moreover, researchers in computing education often have interdisciplinary backgrounds and form teams to aggregate expertise around a research question. Perhaps the most important shared feature between DBR and computing education research in general is that it is conducted in authentic learning environments with students who are learning computing.

Given that computing education research is conducted in authentic learning environments, often computing courses, it is important that the community recognize the difference between DBR and Scholarship of Teaching and Learning (SOTL). SOTL is the practice of integrating teaching with research about teaching (Hutchings & Schulman, 1999). In SOTL, the instructor of a course will test instructional design, tools, and activities within their own courses and collect data on their efficacy. Though this evidence-based approach is reminiscent of DBR, SOTL focuses on advancing the practice of teaching (Bender & Gray, 1999), while DBR balances advancing the practice of teaching with building learning theory.

DBR, therefore, is in the middle of the spectrum between methods that focus on improving practice, such as SOTL, and methods that focus on building theory, such as psychological research. It maintains this balance by employing experimental methodology, but only as far as it fits within the authentic learning environment. In addition, the research team might include the instructor, but the instructor is not the sole researcher in DBR. In DBR, outside perspectives of the learning environment, especially as an intervention is tested in multiple environments, help to distinguish between generalizable features of the intervention and context-specific features. A larger research team also helps implement multiple methods of collecting and analyzing data. A mixed-methods approach (i.e., using both qualitative and quantitative methods and analyses) is common in DBR because it captures multiple aspects of learning and the environment, which is a central feature of learning sciences work.

DBR is a good neutral point for researchers in computing education who are developing the methods for a study. It will not be the best approach for every research question, but its basic tenets are valuable across many types of projects. Of course, research that is more practically driven or more theoretically driven will be more appropriate for some studies, depending on the goals or the strengths of the research team. In any case, though, DBR is a good starting point that will push the community to think about both the rigor and impact of our research.

8.4 Stages of Learning Sciences Projects

We have introduced the learning sciences as a field that embraces constructivist pedagogies, values holistic exploration of learning environments, and engages in design innovation in systematic ways to understand and coevolve

educational interventions. This often means that **empirical** projects in the learning sciences are made up of four stages: (1) conducting studies to better understand a learning context and its learners; (2) designing initial interventions based on these findings; (3) iterating on the designs based on lessons learned during empirical trials; and (4) scaling up a well-tested intervention beyond the local context in which it was refined to contribute to theory. In this section, we will highlight three recent research projects that are comfortably situated between the learning sciences and computing education communities while also exemplifying these core stages.

Stage 1: Prior to designing something, learning scientists engage in studies that aim to inform the intervention, regardless of whether that intervention is a piece of educational software, a set of classroom activities, or an entire informal learning environment. In addition to a thorough literature search, this exploration includes qualitative and/or quantitative studies that uncover details about the targeted content, the attitudes and dispositions of learners, and other sociotechnical elements in the learning environment. These studies are often motivated by observable opportunities in the world, but might also be theoretically driven.

DiSalvo's early work on the Glitch project illustrates many of these elements. Her work initially sought to explore the relationship between videogame play and participation in undergraduate computing majors, with a particular eye toward differences connected to learners' race and gender (DiSalvo & Bruckman, 2009). This early work identified a curious pattern that young black and Hispanic men were the most frequent game players, despite being traditionally underrepresented in computing fields. Additional studies (DiSalvo & Bruckman, 2010; DiSalvo, Crowley, & Norwood, 2008; DiSalvo, Yardi, & Bruckman, 2011) about the unique gaming attitudes, play practices, and cultural values of young African American men directly shaped the initial Glitch Game Tester program. The intervention competitively engaged participants as beta-testers for forthcoming games related to their passions while learning about programming so that they could provide more actionable bug reports to the professional developers (DiSalvo et al., 2013). The extensive formative work to understand the important variables and opportunities for the learners was a crucial element in ensuring the experience was both effective and perceived as authentic.

Stage 2: Having distilled insights about the design space from prior literature and formative studies, the learning scientist then seeks to reify these observations in a way that will positively impact one or more aspects of the learning environment. Consistent with DBR practices, initial interventions are often collaboratively devised by teams of researchers and teachers with the intent of being deployed in a particular educational context. Generalizability is not of great concern at this stage, but rather the goal is to pilot a proof of concept for the intervention and explore the pros and cons of its affordances for learning.

The work of Shapiro and colleagues on BlockyTalky provides a helpful example of this second stage. Their initial explorations around learning

environments involving creative engineering tasks raised questions about what and how we assess student learning in these settings (Deitrick, O'Connell, & Shapiro, 2014). They then created a programming environment and physical computing platform called BlockyTalky, which provide a rich toolkit for the creation of projects using distributed computational elements while also abstracting away technical details like network protocols and explicit data transfer between computational nodes. The initial version was piloted in a computer music summer camp experience in which middle school students designed and built novel musical instruments (Deitrick et al., 2015; Shapiro et al., 2017). These early deployments of BlockyTalky sought to explore both the affordances of the technical system and also the rich creative and social learning environment in which it was used. For example, distributed cognition theory (Deitrick et al., 2015) was used as a lens to understand how knowledge is shared, offloaded, and transformed in the classroom.

Stage 3: As indicated in the discussion of DBR, learning scientists regularly engage in the iterative refinement of an intervention in order to "get it right" in the desired context. Rarely (if ever) do initial deployments meet all of the design objectives, and both empirical data and formative feedback from project stakeholders are vital in making improvements. At this stage, the context of the investigation might still be quite limited, focusing on a single teacher's class or a single institution. Alternatively, researchers might explore deployment in a handful of carefully selected contexts. The overarching goal is to carefully hone the intervention over time while collecting evidence about its effectiveness in particular contexts. Some efforts also engage in initial theory-building work at this stage so as to begin rising above a particular context.

Continuing the prior example, work on BlockyTalky demonstrates how the research team has iterated to expand the system's capabilities and its applicability to a much wider range of making tasks beyond musical computing. Subsequent iterations explored its use in more general makerspace classrooms while also refining the pedagogical practices and classroom routines employed while students create their projects (Deitrick, Shapiro, & Gravel, 2016; Kelly et al., in press). Formative feedback from learners and facilitators highlighted important and unexpected downsides of abstracting network communication, and new versions of BlockyTalky feature language structures that are more explicit about the flow of control between decentralized components to help learners better understand where and when their code is executing (Kelly et al., in press). At the same time, the researchers have identified opportunities to provide BlockyTalky integration with the rich set of computational tools that were also being used by their partner teachers and students, thus spurring ongoing development work.

Stage 4: Once learning scientists have found an effective intervention, they turn their attention to generalization. Generalization can take many forms, such as scaling up in similar learning contexts or replicating results with new populations, content, or learning environments. This stage is critical to building

theory, but it is often overlooked in learning sciences and computing education research. Many times we do a great job with the first three stages, building a great tool or designing an effective learning environment, but then we move on to the next project before we test the boundaries. This last stage, however, is crucial to the science of learning sciences.

Margulieux and Morrison's work on subgoal labels illustrates one way to approach generalization. Margulieux started exploring the efficacy of subgoal-labeled worked examples in undergraduate programming education in a lab, not a classroom, in order to carefully control the learning environment and learners' prior knowledge (i.e., stage 3) (Margulieux, Guzdial, & Catrambone, 2012). Because participants in the lab would not know anything about programming, she used a block-based programming language and gave participants 30 minutes of instruction at a time, neither of which are typical in undergraduate programming education. When the results of these lab studies were positive, Morrison recommended that they try out subgoal-labeled worked examples in a more authentic environment with real introduction to programming students (Morrison, Margulieux, & Guzdial, 2015). They then went on to replicate their results at three universities, each with unique characteristics, to ensure that their findings would replicate with new populations (Margulieux et al., 2016; Morrison, Decker, & Margulieux, 2016). Morrison, Decker, and Margulieux are continuing their work at the time of writing this chapter to develop subgoal-labeled worked examples for an entire introduction to programming curriculum and to evaluate their efficacy in universities across the USA. By doing so, they are contributing to theory on subgoal learning, cognitive load, and development of problem-solving skills.

8.5 Conclusion

Learning sciences and computing education research have a lot in common, yet the two communities can still learn from each other and benefit from a closer relationship. Computing education researchers should draw from the rich literature, general learning theories, and methodologies that the learning sciences offers to guide our research, increasing the rigor within our community and contributing to education outside of computing. In turn, computing education research can inform big challenges in education, such as cognitive development of skill in other technical and complex disciplines, sociocultural development of identity in disciplines with underrepresentation issues, integration of computing and computational thinking throughout other disciplines, and much more. Perhaps the most pressing problem right now is that computing is becoming ubiquitous, but computing education is not. More connections between computing education researchers and learning scientists would contribute to the integration of computing into the education of other disciplines. Integration with other disciplines not only increases the impact of our work, but also serves to increase computing literacy for everyone.

References

Alexander, P. A., Schallert, D. L., & Reynolds, R. E. (2009). What is learning anyway? A topographical perspective considered. *Educational Psychologist*, 44(3), 176–192.

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.

Anderson, J. R. (1996). ACT: A simple theory of complex cognition. *American Psychologist*, 51(4), 355.

Barab, S. (2014). Design-based research: A methodological toolkit for engineering change. In R. Keith Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences,* 2nd edn. (pp. 151–170). Cambridge, UK: Cambridge University Press.

Barab, S., & Squire, K. (2004). Design-based research: Putting a stake in the ground. *The Journal of the Learning Sciences*, 13(1), 1–14.

Ben-David Kolikant, Y., & Ben-Ari, M. (2008). Fertile zones of cultural encounter in computer science education. *The Journal of the Learning Sciences*, 17(1), 1–32.

Bender, E., & Gray, D. (1999). The scholarship of teaching. *Research and Creative Activity,* 12(1). Retrieved from www.indiana.edu/~rcapub/v22n1/p03.html

Blackwell, A. F. (2002). First steps in programming: A rationale for attention investment models. In *Human Centric Computing Languages and Environments* (pp. 2–10). Piscataway, NJ: IEEE.

Blackwell, L. S., Trzesniewski, K. H., & Dweck, C. S. (2007). Implicit theories of intelligence predict achievement across an adolescent transition: A longitudinal study and an intervention. *Child Development,* 78(1), 246–263.

Brown, A. L. (1992). Design experiments: Theoretical and methodological challenges in creating complex interventions in classroom settings. *The Journal of the Learning Sciences*, 2(2), 141–178.

Bruner, J. S. (1973). Beyond the information given. In J. M. Anglin (Ed.), *Beyond the Information Given: Studies in the Psychology of Knowing* (pp. 143–175). New York: W.W. Norton & Company.

Cobb, P., Confrey, J., DiSessa, A., Lehrer, R., & Schauble, L. (2003). Design experiments in educational research. *Educational Researcher*, 32(1), 9–13.

Collins, A. (1992). Toward a design science of education. In *New directions in Educational Technology* (pp. 15–22). Berlin, Germany: Springer.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. *Knowing, Learning, and Instruction: Essays in Honor of Robert Glaser*, 18, 32–42.

Collins, A., & Kapur, M. (2014). Cognitive apprenticeship. In R. Keith Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences,* 2nd edn. (pp. 109–127). Cambridge, UK: Cambridge University Press.

Cooper, S., & Cunningham, S. (2010). Teaching computer science in context. *ACM Inroads*, 1(1), 5–8.

Design-Based Research Collective (2003). Design-based research: An emerging paradigm for educational inquiry. *Educational Researcher*, 32(1), 5–8.

Deitrick, E., O'Connell, B., & Shapiro, R. B. (2014). The discourse of creative problem solving in childhood engineering education. In *Proceedings of the International Conference of the Learning Sciences* (pp. 591–598). Boulder, CO: ISLS.

Deitrick, E., Shapiro, R. B., Ahrens, M. P., Fiebrink, R., Lehrman, P. D., & Farooq, S. (2015). Using distributed cognition theory to analyze collaborative computer science learning. In *Proceedings of the Eleventh Annual Conference on International Computing Education Research* (pp. 51–60). New York: ACM.

Deitrick, E., Shapiro, R. B., & Gravel, B. (2016). How do we assess equity in programming pairs? Singapore. In *Proceedings of the International Conference of the Learning Sciences* (pp. 370–7). Singapore: ISLS.

DiSessa, A. A. (1991). Local sciences: Viewing the design of human–computer systems as cognitive science. In J. M. Carroll (Ed.), *Designing Interaction: psychology at the human-computer interface* (pp. 162–202). Cambridge, UK: Cambridge University Press.

DiSalvo, B. J., & Bruckman, A. (2009). Questioning video games' influence on CS interest. In *Proceedings of the 4th International Conference on Foundations of Digital Games* (pp. 272–278). New York: ACM.

DiSalvo, B., & Bruckman, A. (2010). Race and gender in play practices: Young African American males. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games* (pp. 56–63). New York: ACM.

DiSalvo, B. J., Crowley, K., & Norwood, R. (2008). Learning in context: Digital games and young black men. *Games and Culture,* 3(2), 131–141.

DiSalvo, B., Guzdial, M., Meadows, C., Perry, K., McKlin, T., & Bruckman, A. (2013). Workifying games: Successfully engaging African American gamers with computer science. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education* (pp. 317–322). New York: ACM.

DiSalvo, B., Yardi, S., Guzdial, M., McKlin, T., Meadows, C., Perry, K., & Bruckman, A. (2011). African American men constructing computing identity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2967–2970). New York: ACM.

Eccles, J. S. (1983). Expectancies, values, and academic behaviors. In J. T. Spence (Ed.), *Achievement and Achievement Motives* (pp. 75–146). San Francisco, CA: Freeman.

Eccles, J. S. (1987). Gender roles and women's achievement-related decisions. *Psychology of Women Quarterly*, 11(2), 135–172.

Edelson, D. C. (2002). Design research: What we learn when we engage in design. *The Journal of the Learning Sciences,* 11(1), 105–21.

Engeström, Y. (1987). *Learning by Expanding*. Cambridge, UK: Cambridge University Press.

Engeström, Y., & Sannino, A. (2010). Studies of expansive learning: Foundations, findings and future challenges. *Educational Research Review*, 5(1), 1–24.

Esmonde, I. (2017). Power and sociocultural theories of learning. In I. Esmonde & A. Booker (Eds.), *Power and Privilege in the Learning Sciences: Critical and Sociocultural Theories of Learning* (p. 6). New York: Routledge.

Greeno, J. G., & Engeström, Y. (2014). Learning in activity. In R. Keith Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences*, 2nd edn. (pp. 128–150). Cambridge, UK: Cambridge University Press.

Guzdial, M. (2003). A media computation course for non-majors. *ACM SIGCSE Bulletin*, 35(3), 104–108.

Guzdial, M. (2010). Does contextualized computing education help? *ACM Inroads*, 1(4), 4–6.

Guzdial, M., & Tew, A. E. (2006). Imagineering inauthentic legitimate peripheral participation: An instructional design approach for motivating computing education.

In *Proceedings of the Second International Workshop on Computing Education Research* (pp. 51–58). New York: ACM.

Hermes, M., Bang, M., & Marin, A. (2012). Designing Indigenous language revitalization. *Harvard Educational Review*, 82(3), 381–402.

Hmelo-Silver, C. E., Duncan, R. G., & Chinn, C. A. (2007). Scaffolding and achievement in problem-based and inquiry learning: A response to Kirschner, Sweller, and Clark. *Educational Psychologist*, 42(2), 99–107.

Hoadley, C. M. (2004). Methodological alignment in design-based research. *Educational Psychologist*, 39(4), 203–212.

Hulleman, C. S., & Harackiewicz, J. M. (2009). Promoting interest and performance in high school science classes. *Science*, 326, 1410–1412.

Hutchings, P., & Shulman, L. S. (1999). The scholarship of teaching: New elaborations, new developments. *Change: The Magazine of Higher Learning*, 31(5), 10–15.

Kafai, Y. B. (1995). *Minds in Play*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Kafai, Y. B., & Burke, Q. (2014). *Connected Code: Why Children Need to Learn Programming*. Cambridge, MA: MIT Press.

Kafai, Y. B., Lee, E., Searle, K., Fields, D., Kaplan, E., & Lui, D. (2014). A crafts-oriented approach to computing in high school: Introducing computational concepts, practices, and perspectives with electronic textiles. *ACM Transactions on Computing Education (TOCE)*, 14, 1.

Kaptelinin, V., & Nardi, B. A. (2006). *Acting with Technology: Activity Theory and Interaction Design*. Cambridge, MA: MIT Press.

Kapur, M. (2008). Productive failure. *Cognition and Instruction*, 26(3), 379–424.

Kapur, M. (2014). Productive failure in learning math. *Cognitive Science,* 38(5), 1008–1022.

Kapur, M. (2015). The preparatory effects of problem solving versus problem posing on learning from instruction. *Learning and Instruction*, 39, 23–31.

Kapur, M., & Bielaczyc, K. (2012). Designing for productive failure. *Journal of the Learning Sciences*, 21(1), 45–83.

Kelleher, C., Pausch, R., & Kiesler, S. (2007). Storytelling Alice motivates middle school girls to learn computer programming. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1455–1464). New York: ACM.

Kelly, A., Finch, L., Bolles, M., & Shapiro, R.B. (2018). BlockyTalky: New programmable tools to enable students learning networks. *International Journal of Child-Computer Interaction*, 18, 8–18.

Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, 41(2), 75–86.

Kolodner, J. L. (2004). The learning sciences: Past, present, and future. *Educational Technology: The Magazine for Managers of Change in Education,* 44(3), 37–42.

Lave, J., & Wenger, E. (1991). *Situated Learning: Legitimate Peripheral Participation*. Cambridge, UK: Cambridge University Press.

Lishinski, A., Good, J., Sands, P., & Yadav, A. (2016). Methodological rigor and theoretical foundations of CS education research. In *Proceedings of the 2016 ACM Conference on International Computing Education Research* (pp. 161–169). New York: ACM.

Malmi, L., Sheard, J., Bednarik, R., Helminen, J., Kinnunen, P., Korhonen, A., … Taherkhani, A. (2014). Theoretical underpinnings of computing education

research: what is the evidence? In *Proceedings of the Tenth Annual Conference on International Computing Education Research* (pp. 27–34). New York: ACM.

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., & Resnick, M. (2004). Scratch: A sneak preview [education]. In *Creating, Connecting and Collaborating through Computing.* (pp. 104–109). Piscataway, NJ: IEEE.

Margolis, J., & Fisher, A. (2002). *Unlocking the Clubhouse*. Cambridge, MA: MIT Press.

Margolis, J., Estella, R., Goode, J., Holme, J., & Nao, K. 2008. *Stuck in the Shallow End: Education, Race, and Computing.* Cambridge, MA: MIT Press.

Margulieux, L. E., Guzdial, M., & Catrambone, R. (2012). Subgoal-labeled instructional material improves performance and transfer in learning to develop mobile applications. In *Proceedings of the Ninth Annual International Conference on International Computing Education Research* (pp. 71–78). New York: ACM.

Margulieux, L., Morrison, B. B., Guzdial, M., & Catrambone, R. (2016). Training learners to self-explain: Designing instructions and examples to improve problem solving. In *Proceedings of the International Conference of the Learning Sciences* (pp. 98–105). Singapore: ISLS.

Morrison, B. B., Decker, A., & Margulieux, L. E. (2016). Learning loops: A replication study illuminates impact of HS courses. In *Proceedings of the Twelfth Annual International Conference on International Computing Education Research* (pp. 221–330). New York: ACM.

Morrison, B. B., Margulieux, L. E., & Guzdial, M. (2015). Subgoals, context, and worked examples in learning computing problem solving. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research* (pp. 21–29). New York: ACM.

Nathan, M. J., Rummel, N., & Hay, K. E. (2016). Growing the learning sciences: Brand or big tent? Implications for graduate education. In M. A. Evans, M. J. Packer, & R. K. Sawyer (Eds.), *Reflections on the Learning Sciences* (pp. 191–209). Cambridge, UK: Cambridge University Press.

Nathan, M. J., & Sawyer, R. K. (2014). Foundations of the learning sciences. In R. Keith Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences*, 2nd edn. (pp. 21–43). Cambridge, UK: Cambridge University Press.

Newell, A., & Simon, H. A. (1972). *Human Problem Solving*. Englewood Cliffs, NJ: Prentice-Hall.

Orton, K., Weintrop, D., Beheshti, E., Horn, M., Jona, K., & Wilensky, U. (2016). Bringing computational thinking into high school mathematics and science classrooms. In *Proceeding of the International Conference of the Learning Sciences* (pp. 705–712). Singapore: ISLS.

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books, Inc.

Pea, R. D. (1994). Seeing what we build together: Distributed multimedia learning environments for transformative communications. *The Journal of the Learning Sciences*, 3(3), 285–299.

Porter, L., Guzdial, M., McDowell, C., & Simon, B. (2013). Success in introductory programming: What works? *Communications of the ACM,* 56(8), 34–36.

Resnick, M., & Ocko, S. (1990). *LEGO/LOGO – Learning through and about Design*. Cambridge, MA: Epistemology and Learning Group, MIT Media Laboratory.

Robins, A. (2015). The ongoing challenges of computer science education research. *Computer Science Education,* 25(2), 115–119.

Sannino, A., Daniels, H., & Gutiérrez, K. (Eds.) (2009). *Learning and Expanding with Activity Theory*. Cambridge, UK: Cambridge University Press.

Sawyer, R. K. (2014). The future of learning: Grounding educational innovation in the learning science. In R. Keith Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences,* 2nd edn. (pp. 1–19). Cambridge, UK: Cambridge University Press.

Schmidt, H. G., Loyens, S. M., Van Gog, T., & Paas, F. (2007). Problem-based learning is compatible with human cognitive architecture: Commentary on Kirschner, Sweller, and Clark. *Educational Psychologist*, 42(2), 91–97.

Searle, K. A., & Kafai, Y. B. (2015). Boys' needlework: Understanding gendered and indigenous perspectives on computing and crafting with electronic textiles. In *Proceedings of the Eleventh Annual Conference on International Computing Education Research* (pp. 31–39). New York: ACM.

Shapiro, R. B., Kelly, A., Ahrens, M., Johnson, B., Politi, H., & Fiebrink, R. (2017) Tangible distributed computer music for youth. *The Computer Music Journal*, 41(2), 52–68.

Sommerhoff, D., Szameitat, A., Vogel, F., Chernikova, O., Loderer, K., & Fischer, F. (2018). What do we teach when we teach the learning sciences? A document analysis of 75 graduate programs. *Journal of the Learning Sciences*, 27(2), 319–351.

Stahl, G. (2005). Group cognition in computer-assisted collaborative learning. *Journal of Computer Assisted Learning*, 21(2), 79–90.

Sweller, J., Kirschner, P. A., & Clark, R. E. (2007). Why minimally guided teaching techniques do not work: A reply to commentaries. *Educational Psychologist*, 42(2), 115–121.

Tan, E., Kang, H. O'Neill, T., & Calabrese Barton, A. (2013). Desiring a career in STEM-related fields: How middle school girls articulate and negotiate between their narrated and embodied identities in considering a STEM trajectory. *Journal of Research in Science Teaching,* 50(10), 1143–1179.

Tobias, S., & Duffy, T. M. (Eds.) (2009). *Constructivist Instruction: Success or Failure?* Abingdon, UK: Routledge.

Turkle, S., & Papert, S. (1990). Epistemological pluralism: Styles and voices within the computer culture. *Signs: Journal of Women in Culture and Society*, 16(1), 128–157.

Vygotsky, L. S. (1978). *Mind in Society*. Cambridge, MA: Harvard University Press.

Wenger, E. (1998). *Communities of Practice: Learning, Meaning, and Identity*. Cambridge, UK: Cambridge University Press.

Wigfield, A. (1994). Expectancy-value theory of achievement motivation: A developmental perspective. *Educational Psychology Review*, 6(1), 49–78.

Wigfield, A., & Eccles, J. S. (2000). Expectancy-value theory of achievement motivation. *Contemporary Educational Psychology*, 25(1), 68–81.

Wigfield, A., Tonks, S., & Klauda, S. L. (2009). Expectancy-value theory. In K. Wentzel & D. Miele (Eds.), *Handbook of Motivation at School* (pp. 55–75). New York: Routledge.

Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2), 89–100.

Yoon, S. A., & Hmelo-Silver, C. E. (2017). What do learning scientists do? A survey of the ISLS membership. *The Journal of the Learning Sciences*, 26(2), 167–183.