Defining Computational Thinking for Science, Technology, Engineering, and Math

David Weintrop, Elham Beheshti, Michael Horn, Kai Orton,
Kemi Jona, Laura Trouille, Uri Wilensky

Abstract: With the inclusion of "Computational Thinking Skills" in the Next Generation Science Standards, a new urgency has come to the challenge of defining what exactly is meant by computation thinking (CT). In response to this challenge, we propose a definition for CT in STEM that draws on existing CT and STEM literature and is grounded in authentic CT in STEM practices. Our definition takes the form of a skills taxonomy that breaks CT in STEM into four main categories: Data and Information Skills, Modeling and Simulations Skills, Computational Problem Solving Skills, and Systems Thinking Skills. In this paper, we present our taxonomy along with a broader motivation for this work and a discussion of the procedure followed to produce our definition.

The release of the Next Generation Science Standards (NGSS) has been the source of both excitement and trepidation among science teachers. Excitement in the shift from old standards based on concept inventories to new standards that emphasize scientific practice and investigation; and trepidation in the recognition that with these new standards comes the uncertainty of integrating them into existing classroom routines. The NGSS outline eight distinct practices. While some of these practices are familiar to veteran teachers, such as "planning and carrying out investigations", others are less familiar, specifically the practice of "using mathematics and computational thinking". With the growing importance of computation in science, it seems appropriate that there be practices that coincide with the emergence and use of new technologies. The difficulty is that practices collected under the umbrella term "computational thinking" (National Research Council [NRC], 2010; Papert, 1980; Wing, 2006), have not been clearly defined in terms of their use in science, technology, engineering, and math (STEM) classrooms. It is this issue that we seek to address with this paper.

The term, "computational thinking" (CT), has been the source of much debate and discussion in the field of computer science education and education more broadly (NRC, 2011). Much of the discussion has revolved around the central question of "What is Computational Thinking?" More nuanced forms of this question include: how is it different than other forms of thinking like mathematical thinking, algorithmic thinking, or problem solving more generally? How does it relate to the field of computer science? And, does it include computer programming and does it always require a computer? Trying to answer these questions for the broader academic community is beyond the scope of this effort. We restrict our analysis in two dimensions. First, instead of CT sui

generis, we focus on CT as it applies to the STEM disciplines. Second, we focus on the needs of in-service teachers who are expected to integrate CT practices into their classrooms as early as next year. by presenting a concrete definition for the skills that comprise computational thinking in STEM (CT-STEM).

Our strategy for doing so is not to present a definition in the form of a sentence or two, but instead to categorize the constituent skills that make up this critical scientific practice. Towards this end, we have developed a CT-STEM skills taxonomy to map out the territory that we think is encompassed by CT within STEM disciplines. The taxonomy is grounded in the practice of using CT in professional and educational STEM endeavors. The paper continues with a brief discussion of the motivation for this work and a presentation of the methods used to create the taxonomy before presenting the taxonomy itself.

Why Bring Computational Thinking to STEM?

Beyond the inclusion of CT as a central scientific practice as defined by the NGSS, we think there are other important reasons to introduce CT-STEM to classrooms that have traditionally not involved computation. CT includes a set of skills that are applicable to a broad range of problems and settings. While this is a strength of the skills, it also presents challenges to teaching them, as they are not tied to a specific domain. By fusing CT instruction with STEM disciplines, students can explore and apply CT skills within a more established and accessible STEM context. In this way, STEM can enhance CT learning. Research has also suggested that the reverse is true; CT and the use of computational tools has been shown to enable deeper learning of STEM content areas for students (Guzdial, 1994; National Research Council, 2011; Repenning, Webb, & Ioannidou, 2010; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013; Wilensky & Reisman, 2006).

Another benefit of embedding CT in STEM classrooms is that it allows CT to reach a wider audience than would be possible if it was taught independently. We see three main reasons for this. First, all schools have courses covering the STEM disciplines. By embedding CT in STEM curricula we can directly address issues of schools lacking the resources to have a course dedicated solely to CT. These limiting resources can be qualified CT teachers, dedicated CT classrooms, or availability in student or teacher schedules for a new class. The CT-STEM approach addresses issues of adoption and implementation that have hampered other efforts to introduce new computational subjects into the classroom. Second, the CT-STEM approach can ensure a wide audience is exposed to CT skills, as all students are required to take STEM courses. This directly addresses issues of students self-selecting into (or out of) CT classes, which has been a challenge plaguing computer science education (Margolis & Fisher, 2003; Margolis, 2008). Finally, embedding CT in STEM coursework can address the issues of practicality of implementation, especially with teachers' comfort with the material. In this approach,

the CT skills that are new to teachers are embedded within concepts that teachers already have mastery over, instead of requiring the teachers to learn entirely new concepts.

Origins of the CT-STEM Taxonomy

Rather than defining CT-STEM through a top-down consensus-making approach, we worked with teachers, curriculum developers, and STEM professionals to identify characteristics of CT-STEM grounded in existing practices. We drew on three main resources: exemplary CT-STEM classroom activities; existing CT and STEM concept inventories and standards documents; and interviews with STEM professionals that use CT in their work. This approach was designed to ensure validity of the taxonomy.

The first step in the creation of our CT-STEM taxonomy was to review existing CT literature (for example: Barr & Stephenson, 2011; Brennan & Resnick, 2012; Grover & Pea, 2013; Lee et al., 2011; NRC, 2010, 2011; Wing, 2006). While researchers had differing opinions on the details of CT, there were common skills and practices that were repeatedly cited as being central, such as the use of abstractions and the creation of algorithms. Since we see CT as being heavily informed by the fields of computer science and engineering, we also gathered and analyzed computer science, engineering, and technology content frameworks (such as the new AP CS Principles course, and the NAEP Technology and Engineering Literacy Framework). In reviewing these documents, our goal was to map out what the existing literature identified as central to CT and begin to gather candidate skills for inclusion in our taxonomy. It is worth mentioning that throughout this portion of the process we did not incorporate STEM specific ideas or concepts, instead we focused on CT broadly in an effort to understand the broad landscape of CT before narrowing our focus on CT-STEM specifically. This refinement came in the next phase of our development process.

The second source of data for the creation of the CT-STEM taxonomy was collecting and open coding of STEM activities designed to introduce CT into the classroom. These activities were drawn from a variety of sources including online repositories such as Google's CT initiative; curricular modules developed by recognized leaders in the development of computational activities for STEM classrooms including the Center for Connected Learning and Computer-Based Modeling and Concord Consortium; and classroom activities developed as part of the "Reach for the Stars" project that links STEM graduate students with high school teachers to develop classroom-ready activities based on their research. These materials were collected and open coded by a pair of researchers, looking to identify CT practices. Two researchers independently coded each classroom activity. The two sets of codes were then merged and reorganized into high-level themes. These themes became the four categories of the taxonomy.

The final source of data for our CT-STEM skills taxonomy was the interviews conducted with professionals from STEM fields whose daily work relies heavily on computation and CT practices. During these interviews, the CT-STEM experts were

asked to explain the nature of their work, the role computers and CT in their profession, and articulate the practices they found most useful in taking advantage of computational power. These interviews were transcribed and analyzed as a way to validate the taxonomy and ground it in authentic CT-STEM practices.

The CT in STEM Skills Taxonomy Framework

The taxonomy is broken down into four major categories: Data and Information Skills, Modeling and Simulation Skills, Computational Problem Solving Skills, and Systems Thinking Skills. Each of these categories is composed of a subset of five to seven skills. In this section we present each category, give a brief description, and list each of its constituent skills. The full version of this paper will more fully discuss each category and provide a more detailed description of each skill.

Data and Information Skills

Data play a critical role in STEM fields. They serve many purposes, take many forms, and play a variety of roles in the course of conducting STEM inquiry. There are many skills associated with collecting and using data effectively and efficiently. Many of these skills overlap with CT skills; together, CT skills and data skills empower learners to ask and answer challenging STEM questions. CT skills are used in all facets of data-related STEM work from the initial data collection phase all the way through drawing conclusions and sharing findings.

The skills that make up the Data and Information Skills category are: *Collecting Data, Creating Data, Manipulating Data, Analyzing Data,* and *Visualizing Data.*

Modeling and Simulations Skills

The use of computational models is a central practice in the investigation of STEM phenomena. We use the term computational models to refer to non-static computer-based simulations or models of real world phenomena. These tools make it possible to investigate questions and test hypotheses that would otherwise be too expensive, too dangerous, too difficult or entirely not possible to carry out otherwise. Computational tools make the practice of modeling and simulation possible on a scale that is not possible otherwise. However, one must keep in mind that computational models are not the real world and therefore be aware of the limitations of such tools.

The skills that make up the Modeling and Simulation Skills category are: *Using Computational Models to Understand a Concept, Understanding How and Why Computational Models Work, Assessing Computational Models, Using Computational Models to Find and Test Solutions,* and *Building New/Extending Existing Computational Models.*

Computational Problem Solving Skills

Problem solving is an integral part of STEM fields. The process of researching and developing understandings of STEM phenomena is full of problems and challenges

that must be overcome. One important aspect of CT-STEM is the ability to effectively take advantage of computational power in the pursuit of knowledge and understanding by employing computational tools and strategies. Using problem solving strategies while working in computational contexts is a critical skill for harnessing new technologies in purist of STEM endeavors. STEM practitioners benefit from developing computational skills that enable them to express ideas in a form that a computer can interpret and execute to investigate STEM phenomena.

The skills that make up the Computational Problem Solving Skills category are: *Troubleshooting and Debugging, Programming, Choosing Effective Computational Tools, Assessing Different Approaches/Solutions to a Problem, Developing Modular Computational Solutions, Using Problem Solving Strategies,* and *Creating Abstractions.*

Systems Thinking Skills

STEM phenomena rarely involve individual objects acting in isolation; instead they are often the result of the interactions between elements, which constitute a system. These elements could be anything from organisms in an ecosystem to mechanical components of a car engine to chemical elements in a solution. Across these varied situations, there is a set of CT-STEM skills that are useful for identifying different elements of a system and understanding how they function and interact. These skills include being able to reason about the system as a whole as well as about the elements and behaviors that constitute the system, and move back and forth between these differing views of the same phenomenon.

The skills that make up the Problem Solving Skills category are: *Investigating a System as a Whole, Understanding the Relationships within a System, Thinking in Levels, and Visualizing Systems,* and *Identifying, Understanding and managing Complexity.*

In the full version of this paper, we will further elaborate each of the sections briefly presented herein, including a more fully developed motivation and theoretical framework, and an expanded, more detailed version of the CT-STEM taxonomy we are proposing.

References

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *ACM Inroads*, *2*(1), 48–54.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. Presented at the American Education Researcher Association, Vancouver, Canada. Retrieved from http://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf

Grover, S., & Pea, R. (2013). Computational Thinking in K-12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38–43. doi:10.3102/0013189X12463051

Guzdial, M. (1994). Software-Realized Scaffolding to Facilitate Programming for Science Learning. *Interactive Learning Environments*, *4*(1), 001–044. doi:10.1080/1049482940040101

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., … Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, *2*(1), 32–37.

Margolis, J. (2008). *Stuck in the shallow end: Education, race, and computing*. The MIT Press.

Margolis, J., & Fisher, A. (2003). *Unlocking the clubhouse: Women in computing*. The MIT Press.

National Research Council. (2010). *Report of a Workshop on The Scope and Nature of Computational Thinking*. Washington, D.C.: The National Academies Press.

National Research Council. (2011). *Report of a Workshop of Pedagogical Aspects of Computational Thinking*. Washington, D.C.: The National Academies Press.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic books.

Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 265–269).

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 1–30.

Wilensky, U., & Reisman, K. (2006). Thinking Like a Wolf, a Sheep, or a Firefly: Learning Biology Through Constructing and Testing Computational Theories—An Embodied Modeling Approach. *Cognition and Instruction*, *24*(2), 171–209.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35.