Assessing Computational Thinking

Roxana Hadad
Northeastern Illinois University, USA & University of Illinois at Chicago, USA

Kimberly A. Lawless
University of Illinois at Chicago, USA

INTRODUCTION

Despite the fact that computer science and information technologies have redefined nearly every discipline and are a large part of the current and future economy (Lacey & Wright, 2010), "most states treat high school computer science courses as simply an elective and not part of a student's core education" (Wilson et al., 2010) resulting in the inability to fill many U.S. jobs that will shape our future (National Center for Women and Information Technology, 2009). As a response, there is a call for education, government, and industry to better prepare our students to have the critical thinking skills necessary for developing and interacting with digital devices and information (Wing, 2006; National Governors Association & Pew Center on the States, 2007; Wilson et al., 2010). In her seminal article, Wing (2006) was clear that these "computational thinking" skills should be a part of everyone's education, not just computer science majors. And although *critical thinking* is a large focus of the new Common Core standards (http://www.corestandards.org), it should not be confused with *computational thinking*, or CT. As stated by Voskoglou and Buckley:

It can be concluded that critical thinking is a prerequisite to knowledge acquisition and application to solve problems, but not a sufficient condition when we are faced with complex real technological problems. Technological problems require also a pragmatic way of thinking such as ['computational thinking'.]" (2012, p.32)

The push to develop "computational thinking" skills means we must support students to "apply basic strategies in problem solving, understand the character of a solution or algorithm, and have a sense of the ways in which computerization and digitization have changed how research is conducted," as stated the National Science Foundation's partnership, Mobilize (www.mobilizingcs.org/about/computational-thinking).

While the alarm has been sounded and organizations have mobilized to begin promoting CT in K-12, Wing argues that learning research has yet to be sufficiently utilized to maximize the impact of CT on K-12 education (National Research Council, 2011). This is apparent in the gaps that exist in research on CT assessment (Grover & Pea, 2013), making teaching CT and incorporating it into other domains difficult for K-12 educators. To address this, we used Gagne's outcomes of learning (1977) as a framework for aligning CT objectives with appropriate assessments in a summer robotics program for middle school students at Northeastern Illinois University.

BACKGROUND

Defining Computational Thinking

In a 1996 paper exploring mathematics education, Papert saw computational thinking as a way "to provide a much richer set of new representations of knowledge than the idea of 'procedural representation' that has slipped into cognitive discourse." (http://www.papert.org/articles/AnExplorationintheSpaceofMathematicsEducations.html). The computer was offering new metaphors to think about how humans learn, and how knowledge is stored, accessed, and processed. But Papert saw that beyond being a metaphor, the way that we arranged information for computers could be a way to help K-12 students better organize and operationalize their own thinking. Ten years later, Wing

DOI: 10.4018/978-1-4666-5888-2.ch150

(2006) challenged the education community to bring CT to all students, because it provides problem-solving techniques for addressing our modern problems that is applicable across the curriculum. Since then, NSF has funded the CSTA and ISTE to create teacher resources and toolkits promoting computational thinking (CSTA & ISTE, 2011). Google is also promoting CT throughout the K-12 curriculum by creating and organizing resources for teachers and students (Google, 2013).

While momentum to foster CT is building, in order for the education community to be clear about how to help students develop CT skills, they have to first understand what CT skills are. Wing and her colleagues (2011) define CT as "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can effectively be carried out by an information-processing agent," (e.g. a computer, tablet, phone, etc.) (Wing, 2011). Using this definition, ISTE and CSTA have broken CT into various characteristics, skills, and attitudes, including:

- Formulating problems in a way that enables us to use a computer and other tools to help solve them
- Logically organizing and analyzing data
- Representing data through abstractions, such as models and simulations
- Automating solutions through algorithmic thinking
- Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources
- Generalizing and transferring this problem-solving process to a wide variety of problems
- Confidence in dealing with complexity
- Persistence in working with difficult problems
- Tolerance for ambiguity
- The ability to deal with open-ended problems
- The ability to communicate and work with others to achieve a common goal or solution (Barr, Harrison, & Conery, 2011, p. 21)

At the same time, The College Board, while developing "CS Principles," the Advanced Placement course aimed at promoting CT skills, listed their version of CT "practices":

- Connecting computing
- Developing computational artifacts
- Abstracting
- Analyzing problems and artifacts
- Communicating
- Collaborating (College Board, 2012, p. 2)

Google also entered the discussion, defining CT as involving "a set of problem-solving skills and techniques that software engineers use to write programs that underlie the computer applications you use such as search, email, and maps" (http://www.google.com/edu/computational-thinking/what-is-ct.html). They have broken down CT into four categories:

- Decomposition
- Pattern recognition
- Pattern generalization and abstraction
- Algorithm design

Although these various definitions broadly agree, a concrete definition of CT that can support its broad use in systematic assessment does not seem to exist. The computing and education communities need a common definition of CT in order to measure whether students are acquiring these skills.

Assessing Computational Thinking

The issue of assessment is critical, because assessment not only determines whether or not educational goals are being met (iEARN, 2013), it also drives the design of the curriculum (iEARN, 2013; Wiggins & McTighe, 1998). Grover and Pea make the gravity of the lack of CT assessment clear: "Without attention to assessment, CT can have little hope for making its way successfully into any K-12 curriculum" (2013, p. 41).

Currently, programming environments are most often used as the basis for assessment (Werner, Denner, Campe, & Kawamoto, 2012; Fields, Searle, Kafai, & Min, 2012). Also, the Exploring Computer Science (ECS) curriculum team have begun a Principled Assessment of Computational Thinking (PACT) (CTL, 2013), while the College Board continues the CS Principles AP exam. However, if CT learning and problem solving moves into areas besides programming (Wing, 2006), there needs to be a broader range of opportuni-