Computational Thinking for Undergraduate Students in the Context of Preservice Teacher Education

Submitted by:
Susan Miller, doctoral student Math C&I, School of Education

Faculty Advisor:
David C. Webb, Associate Professor, Math Education, School of Education

1. Introduction/Project Overview

The purpose of this project is to research undergraduate students' conceptions of computational thinking before and after completing computational thinking activities. This research will be accomplished in three steps. First, I plan to develop and administer a survey to undergraduate students to help me understand the ways in which undergraduate students currently conceptualize computational thinking. Second, I plan to develop a set of computational thinking course activities designed for undergraduate students who are pursuing licensure in secondary science and mathematics. These students will use the computational thinking activities as a means to design and explore ways in which technology can be used to solve complex problems through modeling. Specifically, these activities will be designed to supplement existing methods activities through the addition of visual programming skills with the goal of increasing computational thinking skills. Specifically, these activities will encourage undergraduate students to not only be users of simulations, but to be creators of simulations. The goal of these activities will be to support undergraduate students pursuing licensure as a secondary math or science teacher in bringing computational thinking skills to their own classrooms in a variety of ways. Finally, these undergraduate students will again take a survey to describe their conceptions of computational thinking to show ways in which computational thinking develops and changes as a result of additional activities.

This project will enable me to answer these research questions:

- In what ways do undergraduate students currently conceptualize computational thinking?
- In what ways are undergraduate students' conceptualizations of computational thinking malleable as a result of additional undergraduate activities?

2. Background

The need for students who think computationally (that is, students who are able to find ways to use technology to solve more complex problems through modeling) is growing, resulting in increased interest in providing computational thinking skills through computer science education to students from the early grades through high school and college. However, there is little curriculum developed for classrooms. There is also little research in how to support teachers in their efforts. This problem is compounded by the massive curriculum changes occurring in our society with the nearly nationwide adoption of the Common Core State Standards (in both mathematics and language arts) and the soon to follow Next Generation Science Standards, which have already been adopted by 14 states. Both sets of standards require that students are able to use computational thinking skills to create models of real-world phenomena, yet do not explicitly describe ways to provide support for computational thinking. As a result, from elementary to secondary classes, there are few opportunities for students to learn computational thinking practices, and even fewer that offer the opportunity to learn these practices through computer programming (Cuny, 2012; Wilson & Guzdial, 2010). Furthermore, as most state curriculum standards do not mandate computer programming, few universities offer any type of pre-service teacher instruction in computational thinking, leading to teachers who feel unprepared to teach these skills to their students (Barr & Stephenson, 2011; Yadav, Zhou, & Mayfield, 2011).

With greater constraints on curricular choices, we must help teachers to find ways to bring computational thinking into core classes, so that computational thinking is not perceived as an 'extra class' to fit in to the schedule, but rather a necessary skill to learn throughout all classes.

3. Theoretical Framework

3.1 Computational Thinking

Various groups from universities to professional associations have collaborated toward a working definition of CT in the K-12 realm. The Next Generation Science Standards (NGSS) includes computational thinking as one of the **key eight practices** for students, stating that "[i]n both science and engineering, mathematics and computation are fundamental tools for representing physical variables and their relationships. They are used for a range of tasks such as constructing simulations; statistically analyzing data; and recognizing, expressing, and applying quantitative relationships" (NGSS Lead States, 2013, p. Appendix F, pg 10). As part of these practices, students are expected to develop expertise early on in using existing simulations, and later (in high school) in developing their own simulations, determining appropriate assumptions, and organizing and analyzing data sets. This conceptualization therefore moves technology from the role of luxury in a classroom, to a necessity for scientific inquiry. It also moves from watching and using simulations to actually creating the simulations.

Other organizations, including the Computer Science Teachers Association ("Operational Definition for Computational Thinking for K-12 Education," 2011) and the Royal Society (2012) have worked toward defining computational thinking. Grover and Pea (2013) summarized the work around the conceptualization of CT, arguing that abstraction is the keystone of CT. They found that there were nine elements of CT that were consistent among all definitions (Grover & Pea, 2013, pp. 39-40):

- Abstractions and pattern generalizations (including models and simulations)
- Systematic processing of information
- Symbol systems and representations
- Algorithmic notions of flow of control
- Structured problem decomposition (modularizing)
- Iterative, recursive, and parallel thinking
- Conditional logic
- Efficiency and performance constraints
- Debugging and systematic error detection

Some of these features are already incorporated into existing methods classes for preservice teachers, particularly in math and science methods courses. For example, abstraction and pattern generalizations are a focused part of mathematical methods courses, as are symbol systems and representations. On the other hand, using programming to create algorithms, as well as the conditional logic and error detection that is inherent in programming, is not addressed. Furthermore, computational thinking is more than the ability to code. It requires layers of abstraction to nuance the program in ways that more completely model real-world phenomena.

By understanding current teacher conceptions of CT, it is hoped that course activities can build on these conceptions, thus enabling undergraduate students to maximize their ability to think computationally.

3.2 Theories of learning

Data collected in this study will be interpreted using a social constructivist perspective (Palinscar, 2005; Vygotskij, 1972) . Constructivism is a belief that students actively construct knowledge, rather than passively receive knowledge transmitted by the teacher. It further recognizes that knowledge is built on the foundations of what students have already learned. In computer science, and supported through other subjects, research has suggested that taking a social constructivist approach to education can result in stronger learning for students (Ben-Ari, 1998; Meerbaum-Salant, Armoni, & Ben-Ari, 2013; Palinscar, 2005; Simon, 1995). In this project, using a social constructivist approach will impact the design of the activities, in that they will include discussion and sense-making in ways that encourage the students to work together, and provide feedback to one another, to build conceptual understanding. It will also impact the design of the prompts for the survey, in that the prompts will be designed to elicit student articulation of developing conceptual relationships, to better understand how computational thinking is emerging.

4. Literature Review

The need for computational thinking in the classroom was first discussed by Papert (1972); yet, the term Computational Thinking was ultimately popularized more than 30 years later, by Jeanette Wing (2006, p. 33), where she argued that “[c]omputational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child’s analytical ability.” Wing (2006) argues that it should be a basic course for all students, no different from reading, writing and arithmetic. Some higher education programs have already begun this process. For example, Purdue University has created an undergraduate computational thinking course as a means of fulfilling a requirement for all science undergraduates to take a computer science course (Astrachan, Hambrusch, Peckham, & Settle, 2009). In addition, the need for CT for students in K-12 environments has been argued for by various scholars, (Grover & Pea, 2013; Lu & Fletcher, 2009; Perkovi, Settle, Hwang, & Jones, 2010) the Next Generation Science Standards, and the National Science Foundation (e.g., STEM+C, Cyberlearning, and several other solictations).

Preservice teachers who were provided a week long module on computational thinking as part of an existing education class were nearly three times more likely to see computational thinking as a means to solve problems than teachers without this class (Yadav et al., 2011). These results suggest that even a short introduction to computational thinking (two lectures) can influence teacher's conceptions of CT.

While the emphasis has been placed on student acquisition of CT, more work is needed to understand how undergraduate students pursuing licensure in secondary science and mathematics are thinking about CT because teacher's conceptions in other areas in the classroom have been found to significantly influence student conceptions (Thompson, 1992).

5. Methodology

5.1 Course Activities Creation

The creation of computational thinking activities will begin by working to identify possible methods and activities, by researching other existing computational thinking courses offered to undergraduate students as well as the AP Computer Science Practices (AP-CSP) course, which has been developed to teach computational thinking to non-computer science majors. Then, working in conjunction with faculty from both the school of education as well as the computer science department, the CT activities will be developed. The intent of these activities will be both programming experiences as well as non-computer-based computational thinking experiences. Programming activities will use visual drag-and-drop languages, such as Scratch and AgentSheets, to minimize frustration and maximize students' conceptual understanding of algorithmic modeling, and will be oriented toward creating a model of real-world phenomena.

Approval has already been given to incorporate these activities into Dr. David Webb's Perspectives of Mathematics course (EDUC 5317). The activities will be incorporated into other courses as well in order to broaden the range of undergraduate students exposed to the activities.

5.2 Research

This project will enable responses to the following research questions:

- In what ways do undergraduate students currently conceptualize computational thinking?
- In what ways do undergraduate students' conceptualizations of computational thinking develop and change as a result of additional coursework?

As part of this project, a survey tool will be developed using my own prior research of experienced teachers' conceptions of computational thinking as the starting point. This survey will be given to undergraduate students who agree to participate in the research both prior to engaging in the activities, as well as after the completion of the activities, to facilitate understanding of current conceptions of computational thinking as well as how those conceptions change when participating in different computational thinking activities.

This survey tool will include basic demographic data (race, gender) as well as use of technology, prior technology courses, and dispositions towards CS education. The first two categories will be presented as a list of options in which students could select multiple responses and included an option to write in other options. The dispositions category will be administered using a four point Likert scale. Included in this survey will be open-ended questions asking undergraduate students how they would define computational thinking and how computational thinking could be used in a classroom.

6. Project Timeline

To accomplish these goals, the following timeline is proposed.

- Fall 2015: Activities and survey will be created
- Spring 2016: The presurvey will be administered, and the activities will be taught. The post survey will be administered again upon completion of the activities.
- Summer 2016: Results will be analyzed, and the report completed.

The request for IRB approval will be ready to submit as soon as the funding is awarded.

7. Researcher's Expertise

I bring a unique background to this research with an undergraduate degree in chemical engineering, and an MBA. Education was a second career for me, and I have extensive work experience in technical fields including both nuclear weapons production and computer storage solutions, which enabled me to develop strong computational thinking skills. I am finishing my second year in the doctoral program, having completed 53 of the 56 required coursework credits, and will be taking my comprehensive exams in the fall. I have been part of the Scalable Game Design project for 18 months. As part of this project, I have written curricular materials for teachers and have taught professional development classes to more than 60 teachers, focused on using Scalable Game Design to promote computational thinking within classrooms. I have also had two papers published as the primary author from this research (Teachers' Perceptions of Computational Thinking (Miller, Webb, Repenning, & Endo, 2015), and Game Design: Whose Game Works at the End of the Day? (Miller & Webb, 2015), and coauthored others in support of computational thinking (Repenning et al., 2014; Webb & Miller, 2015).

8. Benefits of Research to CU Community and Beyond

The creation of computational thinking activities and the associated research will benefit the educational research community as well as the CU community at large:

- The research will fill a current gap on understanding how undergraduate students are conceptualizing computational thinking. Added to research already published on experienced teachers' conceptions of computational thinking, this research will enable us and other researchers to better understand ways to expand computational thinking throughout classrooms as well as develop professional development activities.
- The computational thinking activities creation will also benefit the CU community by
    - exposing undergraduate students to the goals of computational thinking
    - for students pursuing licensure for secondary math and science education, by providing them with the resources they need, to bring those skills into the classrooms of their students.
    - allowing these pre-developed activities to be used in other classes as well as the Center for STEM Learning, broadening exposure to computational thinking.
- The survey will also be shared with the CU community, enabling research on conceptions of computational thinking to be explored outside of the school of education.

9. References

Astrachan, O., Hambrusch, S., Peckham, J., & Settle, A. (2009). *The present and future of computational thinking*. *Proceedings of the 40th ACM Technical Symposium on Computer Science Education - SIGCSE '09*. doi:10.1145/1508865.1509053

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*, *x*(x), 111–122. Retrieved from http://dl.acm.org/citation.cfm?id=1929905

Ben-Ari, M. (1998). Constructivism in computer science education. *ACM SIGCSE Bulletin*, *30*(1), 257–261. doi:10.1145/274790.274308

Cuny, J. (2012). Transforming high school computing: A call to action. *ACM Inroads, 3*(2), 32. doi:10.1145/2189835.2189848

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

Lu, J. J., & Fletcher, G. H. L. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*. doi:10.1145/1539024.1508959

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (Moti). (2013). Learning computer science concepts with Scratch. *Computer Science Education, 23*(3), 239–264. doi:10.1080/08993408.2013.832022

Miller, S. B., & Webb, D. (2015). Game Design : Whose game works at the end of the day ? In *AERA*. Chicago, IL.

Miller, S. B., Webb, D. C., Repenning, A., & Endo, Y. (2015). Teachers' perceptions of computational thinking. In *American Educational Research Association Conference Proceedings* (pp. 1–8).

NGSS Lead States. (2013). Next Generation Science Standards: For States, by States. *Achieve, Inc. on behalf of the twenty-six states and partners that collaborated on the NGSS.*

Operational Definition for Computational Thinking for K-12 Education. (2011). International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA).

Palinscar, A. S. (2005). Social Constructivist Perspectives on Teaching and Learning. In *An Introduction to Vygotsky* (pp. 346–374).

Papert, S. (1972). Teaching Children Thinking∗. *Innovations in Education & Training International*. doi:10.1080/1355800720090503

Perkovi, L., Settle, A., Hwang, S., & Jones, J. (2010). A Framework for Computational Thinking across the Curriculum. *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education (ITiCS '10)*, 123–127. doi:10.1145/1822090.1822126

Repenning, A., Webb, D. C., Brand, C., Gluck, F., Grover, R., Miller, S. B., … Song, M. (2014). Beyond Minecraft: facilitating computational thinking through modeling and programming in 3D. *IEEE Computer Graphics and Applications*, *34*(3), 68–71.

Simon, M. (1995). Reconstructing mathematics pedagogy from a constructivist perspective. *Journal for Research in Mathematics Education*, *26*(2), 114–145. Retrieved from http://www.jstor.org/stable/749205

Thompson, A. (1992). Teachers' beliefs and conceptions: A synthesis of the research. In *Handbook of research on mathematics teaching and learning* (pp. 127–146).

Vygotskij, L. (1972). Mind in Society: The Development of Higher Psychological Processes. In M. Cole, V. John-Steiner, S. Scribner, & E. Souberan (Eds.), *Thought and Language*. Cambridge, MA: Harvard University Press.

Webb, D. C., & Miller, S. B. (2015). Gender Analysis of a Large Scale Survey of Middle Grades Students ' Conceptions of Computer Science Education. In *Gender IT*. Philadelphia, PA.

Wilson, C., & Guzdial, M. (2010). How to make progress in computing education. *Communications of the ACM*, *53*(5), 35. doi:10.1145/1735223.1735235

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33. doi:10.1145/1118178.1118215

Yadav, A., Zhou, N., & Mayfield, C. (2011). Introducing computational thinking in education courses. *Educational Studies*, 465–470. Retrieved from http://dl.acm.org/citation.cfm?id=1953297