Embedding Computational Thinking in Science, Technology, Engineering, and Math (CT-STEM)

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

In this whitepaper we argue for a radically different strategy for attracting and retaining students in computational sciences. Rather than trying to appeal to students to enroll in a high school computation course directly (a strategy that has not been successful with the previous AP computer science course for the last 20 years – College Board, 2011), we bring computational activities to students by embedding them within STEM coursework that students are already required to take. We believe that engaging students through their existing STEM courses is a strategy that is much more likely to succeed in increasing the interest and appeal of computational thinking.

Still a Nation at Risk

In its 1983 report, *A Nation at Risk*, the National Commission on Excellence in Education responded to the critical lack of qualified Computer Science/STEM workers by sparking the movement to bring technology into our secondary schools. This effort has been largely successful, with technology-rich classrooms at all socioeconomic levels (see Fox 2005). However, this improved access to technology has not been accompanied by curricula emphasizing higher-order computational thinking skills nor sophisticated use of this technology. Criticisms of the standard Computer Science curriculum include its emphasis on basic skills as opposed to problem solving (Goode, Estrella & Margolis 2005) and its exclusive focus on low-level programming (Schofield 1995). Not surprisingly, students perceive the work done by a computer scientist as lacking in creativity, addressing abstract problems rather than issues relevant to everyday life, and as being reserved for an elite group of the "best and brightest", if it is considered at all (Margolis 2008; Stonedahl, Wilkerson, & Wilensky 2011). As a result, while the biggest growth in our 21st century job market is for workers with CT skills, our schools currently produce less than one-third the number of qualified applicants (Levy & Murname 2004). This lack of quality curricula is particularly true for schools dominated by students with low socioeconomic status (SES; Warschauer 2000, 2006). Moreover, research has shown that female and minority students are dissuaded from pursuing CS/STEM education and careers (Margolis et al. 2008; Kozol 1992; Kao 2000; Noguera 2003; NRC 2004). Dominant groups benefit from preparatory privilege (resources and support at home and in school; as well as cultural expectations), which make for an unlevel playing field (Margolis et al. 2008). As a result of conscious and unconscious biases, women and minorities are placed into low-level and feeder CS courses in high school, known for disengaging students (Margolis et al. 2008). Moses & Cobb (2002) label this the civil rights issue of the 21st century, with people of low SES, women, and minorities being the "designated serfs of the information age". These factors have led to the current situation in which only 8% of bachelor degrees and 4% of master degrees in Computer Science are awarded to African-Americans and

Latinos (Zweben 2006). Similarly, women account for only 11% of BAs and 22% of masters degrees in CS (Taulbee 2009).

Casting a Wide Net: Computational Thinking in STEM

We believe that a very promising strategy for addressing the many challenges described above is to embed computational thinking activities in traditional STEM courses. This strategy may very well be the most likely strategy for successfully introducing computational thinking into school settings in the short run. By introducing computational thinking in this way, our goal is to have students apply computational thinking tools and techniques in multiple domains and recognize the utility of computational approaches in a range of applications (NRC 2009). In teaching embedded computational thinking, we avoid repeating what many consider to be a mistake made by math educators in presenting their discipline in its abstract, isolated form. We also directly address student misperception that the work done by people in CS careers lacks creativity, relevance to the real world, and positive impact on people's every day lives (see Jones & Clark 1995; Margolis & Fisher 2002; Jessup & Summer 2005; Yardi & Bruckmann 2007). In our approach, computational thinking is grounded in reality, with real-world application and impact.

CT-STEM Research Agenda

Our proposed research and development work plan is organized around the investigation of the following research questions:

1. *What computational thinking skills are most important for STEM practitioners broadly?*

2. *How can we best teach these skills through curricular units embedded throughout STEM coursework?*

3. *With this approach, how can we increase computational competencies for all students and build interest in computing as a field in its own right?*

4. *Will this approach increase students' interest and engagement in traditional science, technology, engineering, and mathematics disciplines?*

5. *Will this approach prepare students for successful careers in computing-intensive fields?*

6. *Will this approach increase the number of teachers who develop and practice computational methods and approaches?*

Why Embed Computational Thinking in STEM?

The Next Generation Science Standards (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. However, there are other important reasons to introduce computational thinking to classrooms that have traditionally not involved computation. Computational thinking includes a set of skills that are applicable to a

broad range of problems and settings. **By pairing computational thinking instruction with STEM content, students can explore and apply computational approaches within more established and accessible STEM context.** Additionally, by spreading computational thinking skills across the STEM spectrum, students will be exposed to these ideas on different occasions across multiple years and across different content areas. In this way, STEM can enrich computational learning. Research has also shown that the reverse is true; 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). The use of computational tools and computational thinking skills can deepen the learning of STEM content. *This symbiotic relationship is at the heart of our motivation to bring CT and STEM together*.

A second motivation for bringing CT and STEM together is to **reach the widest possible audience**. We see three main reasons for this. First, all schools have existing courses covering the STEM disciplines. By embedding CT in STEM curricula we can directly address issues of schools' inability to have a dedicated computer science or CT course due to a lack of qualified teachers, interested students, and adequate facilities. 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 as part of their high school graduation requirements. This directly addresses issues of students self-selecting into (or out of) CT classes, which has been a challenge plaguing computer science education for many years (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 STEM concepts that teachers already have mastery over, instead of requiring the teachers to learn entirely new concepts.

A final motivation for bringing CT into STEM classrooms is to **better prepare students for the modern landscape of the STEM disciplines**. Computation is an indispensable component of STEM disciplines as they are practiced in the professional world. In the last twenty years, nearly every STEM field has seen the birth or reconceptualization of a computational counterpart, from Computational Engineering and Bioinformatics to Chemometrics and Neuroinformatics. The appearance of "data-driven" computational fields has a rich history intertwined with statistics and dynamic systems theory, but recent advances in high-speed computation and analytical methods have created some of the most powerful tools in understanding phenomena across all spectrum of human inquiry. The 1998 Nobel Prize in Chemistry was awarded to John A. Pople and Walter Kohn for their innovative work in the development of computational methods quantum chemistry. Such a prestigious award hailed the acceptance of computation as a valid and rigorous tool for investigating chemical phenomena. Across research laboratories, engineering and design firms, medical practices, and beyond, computational tools are at the heart of what it means to be a STEM practitioner in the 21st century. Bringing these computational tools and practices into the classrooms gives learners a more realistic view of what STEM fields are as well as better prepares students for STEM careers, a major goal of STEM education

These three features of the CT in STEM approach, paired with the fact that the two are mutually beneficial from an educational perspective, results in a compelling reason for bringing CT and STEM together in the classroom.

References

College Board (2011). Advanced Placement (AP) Exam Data 2011, available at http://professionals.collegeboard.com/data-reports-research/ap/data

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.

Fox, E. (2005). *Tracking U.S. Trends*. Education Week. 24 (35): 40-42.

Goode, J., Estrella, R., & Margolis, J. (2006). *Lost in translation: Gender and high school computer science.* In J.M. Cohoon & W. Aspray (Eds.), *Women and information technology: Research on underrepresentation* (pp. 89-114). Cambridge, MA: MIT Press.

Guzdial, M. (1994). Software-realized scaffolding to facilitate programming for science learning. *Interactive Learning Environments*, *4*(1), 001–044. doi:10.1080/1049482940040101

Guzdial, M. (2008). Paving the way for computational thinking. *Commun. ACM*, *51*(8), 25–27. doi:10.1145/1378704.1378713

Guzdial, M., & Soloway, E. (2003). Computer science is more important than calculus: The challenge of living up to our potential. *SIGCSE BULLETIN*, *35*(2), 5–8.

Jessup, Elizabeth R. & Sumner, Tamara. (2005). *Design-based learning and women's participation in IT.* Frontiers: A Journal of Women Studies, 26(1).

Jones T, Clarke VA (1995). *Diversity as a determinant of attitudes: A possible explanation of the apparent advantage of single–sex settings.* J. Educ. Comput. Res. 12(1): 51-64.

Kao, G. (2000*). Group images and possible selves among adolescents: Linking stereotypes to expectations by race and ethnicity*. Sociological Forum 15 (3): 407-430.

Kozol, J. (1992). *Savage Inequalities.* New York: Harper Perennial.

Levy, F. & Murname, R. (2004). *The new division of labor: How computers are creating the new job market.* Princeton, NJ: Princeton University Press.

Margolis, J., Estrella, R., Goode, J., Jellison Home, J., Nao, K. (2008). *Stuck in the shallow end: Education, race, and computing*. Cambridge Mass: MIT Press.

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

Moses, R., & Cobb, C. (2002). *Radical equations: Civil rights from Mississippi to the Algebra Project*. Boston: Beacon 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.

Noguera, P. (2003). *City Schools and the American Dream.* New York: Teachers College Press.

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.

Settle, A., Franke, B., Hansen, R., Spaltro, F., Jurisson, C., Rennert-May, C., & Wildeman, B. (2012). Infusing computational thinking into the middle- and high-school curriculum. In *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education* (pp. 22–27). New York, NY, USA: ACM. doi:10.1145/2325296.2325306

Settle, A., Goldberg, D. S., & Barr, V. (2013). Beyond computer science: computational thinking across disciplines. In *Proceedings of the 18th ACM conference on Innovation and technology in computer science education* (pp. 311–312). New York, NY, USA: ACM. doi:10.1145/2462476.2462511

Stonedahl, Wilkerson-Jerde, Wilensky, U. (2011). *Re-conceiving Introductory Computer Science Curricula through Agent-based Modeling*. Proceedings of the EduMAS Workshop at AAMAS '09, May 12. Budapest, Hungary. pp. 63-70.

Warschauer, M. (2000). *Technology and School Reform: A view from both sides of the track.* Educational Policy Analysis Archives 8 (4).

Warschauer, M. (2006*). Laptops and literacy: Learning in the wireless classroom.* New York: Teachers College Press.

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.

Wing, J. M. (2011). Research Notebook: Computational Thinking—What and Why? *The Link*, (Spring). Retrieved from http://link.cs.cmu.edu/article.php?a=600

Yardi, S. & Bruckman, A. (2007). *What is computing?: bridging the gap between teenagers' perceptions and graduate students' experiences.* In Anderson, R., Fincher, S., & Guzdial, M. (Eds.), Proceedings of the Third international Workshop on Computing Education Research (ICER '07) (pp. 39-50), Atlanta: ACM, New York, NY.

Zweben, S. (2006). *2004-2005 Taulbee Survey*. Computing Research News, May 2006, 7-17.