

Education

K-12 Computational Learning

Enhancing student learning and understanding by combining theories of learning with the computer's unique attributes.

IN “COMPUTATIONAL THINKING,”¹⁴ Jeannette Wing struck a chord that has resonated strongly (generating positive as well as negative responses) with many computer scientists and non-computer scientists. Wing has subsequently defined computational thinking as the process of abstraction,¹⁵ guided by various engineering-type concerns including efficiency, correctness, and several software engineering “-ilities” (maintainability, usability, modifiability, and so forth). Some have interpreted computational thinking as an attempt to capture the set of computer science skills essential to virtually every person in a technological society, while others view it as a new description of the fundamental discipline that represents computer science and its intersection with other fields. The National Academies report¹ captures both of these views, as well as presenting others.

While we can live with such definitions/descriptions in the higher education arena, we struggle with these notions of computational thinking in the K-12 arena (note that we primarily consider K-12 education within the U.S.). Several concerns spring to mind:

1. Computer science does not appear within the core topics covered in high school. We would have a tough time justifying a computer science course, even the “great ideas” AP Principles course (being developed as part of Denning⁴) replacing Algebra 2, Biology, or American Government. K-12 education is a zero-sum game. If one wishes



Berkeley public elementary school students on a field trip learn how computers enable science.

to add a course, one must also propose a course to be removed.

2. Even as an elective topic, computer science tends to be disproportionately available to those wealthy suburban schools. Margolis et al.⁶ explore this situation in depth within the urban Los Angeles school district.

3. Too few K-12 computing teachers are available to implement a national-scale computing requirement. CUNY’s ambitious 10,000 teacher project² will not produce sufficient numbers of computing teachers required to instruct *all* schoolchildren in the U.S. It would not even get one qualified teacher into each of the nation’s 30,000+ high schools (see http://nces.ed.gov/programs/digest/d09/tables/dt09_086.asp).

4. It is not clear to us how teachers in

other K-12 subjects would take advantage of school children who had been trained in computational thinking.

5. The most common definitions of computational thinking are confusing when explained to non-computer scientists. And many K-12 computing teachers are not computer scientists.

We find the above definitions of computational thinking not especially useful when considered in the context of K-12 education, and, more specifically, K-12 science, technology, engineering, and mathematics (STEM) education. We propose an alternative to the common definition of computational thinking we believe is appropriate for operationalization in K-12 education and consider its broader implications.

A K–12 View of Computational Thinking

We have struggled with how computational thinking might be different from mathematical thinking, algorithmic thinking, quantitative reasoning, design thinking, and several other models of math, science, and even engineering to critical thinking and problem solving. It was after struggling with the latest type of thinking that we realized that perhaps even the term “computational thinking” was misleading (from a K–12 perspective), and we were approaching the definition incorrectly. Rather than considering computational thinking as a part of the process for problem solving, we instead developed a model of computational learning that emphasizes the central role that a computer (and possibly its abstraction) can play in enhancing the learning process and improving achievement of K–12 students in STEM and other courses. The figure here depicts our current working model of computational learning. It should be noted that this model is explicit in its use of a computer and specifically excludes non-cognitive uses of technology (Powerpoint, wikis, blogs, clickers, and so forth).

Similar to Wing’s original vision of computational thinking, we see computational learning as an iterative and interactive process between the human (the K–12 student in our case) and the computer (or, in a more theoretical construct, a model of computation). We also make explicit the two consequences of the human cognitive process, namely, the capacity for abstraction and for problem formulation, and

two strengths of the computer, namely, their ability to present complex data sets, often visually, and their capacity for storing factual and relational knowledge. These four elements frame and establish the boundaries of the iterative interaction between the human being and the computer. Note that the accompanying figure does not explicitly include a teacher, not because we believe teachers are unnecessary, but rather because the role of the teacher in this model is complex and requires further investigation.

In developing this model, we observed that it includes other extant models in scientific learning and inquiry. For example, one can view computational science as the interaction between the human and the computer that is contained within the box where a human being formulates a problem and provides a representation suitable for a computer. The computer then acts on this representation and returns the results of these actions to the human being through, for example, a visual representation. Computational learning expands this interaction by allowing the computer to add foundational knowledge, not just data, unknown to the human and by having the results of the computer’s actions represented in a form compatible with the human’s current capacity for abstraction. In the more interesting instances of computational learning, both of these processes are likely to be adaptive and personalized to the individual.

We make several observations about computational learning:

- ▶ Computational learning is iterative,

requiring interaction between the computer and the human.

- ▶ In computational learning, the computer can compensate for a human’s lack of factual and relational knowledge and mathematical and scientific sophistication.

- ▶ The computer’s ability to quickly compute multiple examples and present them via a modality appropriate to a human’s current development stage and level of meta-cognitive awareness can leverage the human’s inherent, though perhaps not fully conscious, capacity for abstraction.

This model for computational learning differs significantly from other proposed notions of computational thinking. For example, algorithmic thinking does not require a computer and mathematical thinking is almost solely dependent on the human’s formalization capacity for abstraction.²

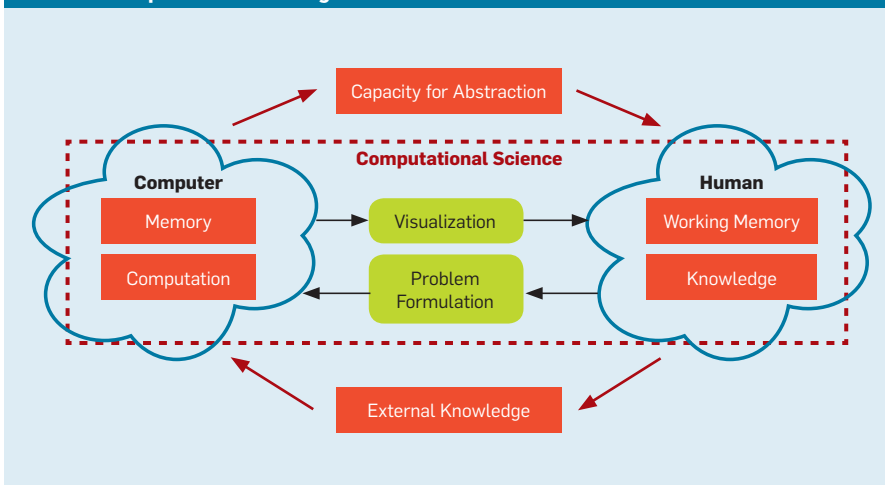
To better understand how our proposed computational learning model can be operationalized, we present two examples: one from middle school computing and one from high school biology.

Digitizing Data

Cutler and Hutton³ modified a CS Unplugged activity on image representation (see http://csunplugged.org/sites/default/files/activity_pdfs_full/unplugged-02-image_representation.pdf) to enable middle school students to work interactively with a computer program as they learn about how computers digitize images. The purpose of these activities is to help students to understand what it means for a computer to digitally represent an image. More importantly, the students learn to move from concrete representations of images to more abstract representations of those images (as a digital representation), and from representation of images in 2D to representing objects in 3D. And this ability to abstract is important across all STEM disciplines.

Students work interactively with the computer program, receiving feedback to their attempts at digitizing their data. Over the series of lessons, they develop an initial ability to abstract away from the physical representation of an image to its digital representation. They are then further able to develop their ability to abstract as they move from 2D

A view of computational learning.



images to 3D objects. Finally, students develop a further ability to abstract. As part of the creation of a single 3D object, say a chair, the students are then challenged to place many chairs in a room. They need to be able to recognize that representing a 3D chair consists of two parts: the relative coordinates of each of the parts of the chair, and the absolute location of one part (say the bottom-right corner of the front-right leg).

Evolutionary Biology

The second example involves the teaching of evolution using computational learning. Our vision is of a 3D visualization system that could simulate evolution. A student could specify an organism, with primitive appendages (arms, legs, joints, and other attributes) to accomplish locomotion. Then, by providing an environment, the student could run the simulation to watch how the organism's ability to move evolves over time as a function of its current locomotion capability coupled with the impact of that organism's environment. Students could change the appendages and/or the environment to observe how such changes lead to a difference in the organism's evolution over time. In computer science terms, this example is similar to passing a program and an initial state as input to a Universal Turing Machine.

Such a simulation allows the student to work interactively with the computer program. The student learns both from the impact of the changes she makes to the initial configuration of the organism and to the initial environment (which will lead to the organism evolving the ability to move differently) as well as by the ability to observe the simulation/visualization as it is running. In science, researchers have found that visualization is central to increasing conceptual understanding and prompting the formation of dynamic mental models of particulate matter and processes (see ^{5,7,9,12}). Visualization and computer interaction through animation allow students to engage more in the cognitive process, and to select and organize more relevant information for problem-solving.⁷ Computer animations incorporated into interactive simulations offer the user a chance to manipulate variables to observe the effect on the system's


This model for computational learning differs significantly from other proposed notions of computational thinking.

behavior (see ^{9,10,13}).

While we know of no tool that provides the exact support/simulation we are describing, there are several available visualization systems that can simulate/model the world. Two of these systems have helped to shape our vision of the above-mentioned simulation: The 3D visualization system, Framsticks (www.framsticks.com) can be used for modeling evolution, and the 2D simulation system, NetLogo (<http://ccl.northwestern.edu/netlogo/>) has many available pre-built simulations, including those that model evolution albeit in a different manner than what we describe.

Conclusion

Most papers we've seen on computational thinking represent attempts at repackaging computing science concepts, especially in the form of algorithmic thinking and introductory programming, sometimes in other domains. Though this may be useful in some contexts, it is unlikely such a simple approach will have significant impact on student learning—of computer science or other disciplines—in the K–12 setting. The proposed model of computational learning combines theories of learning with the computer's superiority in dealing with complexity and variability and its ability to present results using modalities that appeal to the learner in order to enhance student learning and understanding. We believe that computational learning can be framed within various theories of learning, where the computer plays a similar role as Vygotsky's More Knowledgeable

Other (though the computer obviously cannot think at a higher level than the student),¹¹ or even fits within Newell and Simon's Information Processing Theory framework.⁸ Our hope is that by considering our model of computational learning, we can better educate and prepare teachers to benefit from computing in and outside the classroom, and that approaches and computing tools can be identified and built to improve K–12 student STEM learning. 

References

1. Committee for the Workshops on Computational Thinking, National Research Council. *Report of a Workshop on the Scope and Nature of Computational Thinking*. National Academies Press, Washington, D.C., 2010.
2. Cuny, J. Finding 10,000 teachers. *CSTA Voice*, 5, 6 (2010), 1–2; http://www.csta.acm.org/Communications/sub/CSTAVoice_Files/csta_voice_01_2010.pdf
3. Cutler, R. and Hutton, M. Digitizing data: Computational thinking for middle school students through computer graphics. In *Proceedings of the 31st Annual Conference of the European Association for Computer Graphics EG 2010—Education Papers*. (Norrköping, Sweden, May 2010), 17–24.
4. Denning, P. Beyond computational thinking. *Commun. ACM* 52, 6 (June 2009), 28–30.
5. Gilbert, J.K., Justi, R., and Aksela, M. The visualization of models: A metacognitive competence in the learning of chemistry. Paper presented at the 4th Annual Meeting of the European Science Education Research Association, Noordwijkerhout, The Netherlands, 2003.
6. Margolis, J. et al. *Stuck in the Shallow End: Education, Race, and Computing*. The MIT Press, 2008.
7. National Science Foundation. Molecular visualization in science education. Report from the molecular visualization in science education workshop. NCSA access center, National Science Foundation, Arlington, VA, 2001.
8. Newell, A., and Simon, H.A. *Human Problem Solving*. Prentice-Hall, Englewood Cliffs, NJ, 1972.
9. Rotbain, Y., Marbach-Ad, G., and Stavay, R. Using a computer animation to teach high school molecular biology. *Journal of Science Education and Technology* 17 (2008), 49–58.
10. Sewell R., Stevens, R., and Lewis, D. Multimedia computer technology as a tool for teaching and assessment of biological science. *Journal of Biological Education* 29 (1995), 27–32.
11. Vygotsky, L.S. *Mind and Society: The Development of Higher Mental Processes*. Harvard University Press, Cambridge, MA, 1978.
12. Williamson V.M. and Abraham, M.R. The effects of computer animation on the particulate mental models of college chemistry students. *Journal of Research in Science Teaching* 32 (1995), 522–534.
13. Windschitl, M.A. A practical guide for incorporating computer-based simulations into science instruction. *The American Biology Teacher* 60 (1998), 92–97.
14. Wing, J.M. Computational thinking. *Commun. ACM* 49, 3 (Mar. 2006), 33–35.
15. Wing, J.M. Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A*, 366 (2008), 3717–3725.

Stephen Cooper (coopercs@purdue.edu) is an associate professor (teaching) in the Computer Science Department at Stanford University.

Lance C. Pérez (lperez@unl.edu) is an associate professor in the Department of Electrical Engineering at the University of Nebraska-Lincoln where he also holds the position of Associate Vice Chancellor for Academic Affairs.

Daphne Rainey (raineyd4@gmail.com) is a biologist currently serving in a temporary assignment as a program director at NSF's Division of Undergraduate Education within its Education and Human Resources Directorate.

Copyright held by author.