Ubiquity Symposium

The Science In Computer Science

The Computing Sciences and STEM Education

by Paul S. Rosenbloom

Editor's Introduction

In this latest installment of "The Science in Computer Science," Prof. Paul Rosenbloom continues the discussion on whether or not computer science can be considered a "natural science." He argues not only is computing the basis for a true science, it is in fact an entire scientific domain.

Peter Denning
Editor-in-Chief

Ubiquity Symposium

The Science In Computer Science

The Computing Sciences and STEM Education

by Paul S. Rosenbloom

In this short essay, I make three claims:

1. It is time to put the final nails in the coffin of the argument from artificiality, that computing isn't a true science because it studies artificial rather than natural phenomena.
2. It is time to go beyond the straightforward conclusion that computer science is a respectable scientific discipline—such as physics or psychology—to the bolder conclusion that computing actually constitutes an entire domain of science. Let us call this domain the "computing sciences." The computing sciences are the equal of the physical, life and social sciences.
3. The domain of the computing sciences opens up new worlds of subject matter for STEM education.

The first two claims are grounded in a decade of explorations into the nature, structure, stature and role of computing in the sciences. The results of these explorations are recorded in several papers [1, 2], and in the book *On Computing: The Fourth Great Scientific Domain* [3]. The third claim is more speculative, but it follows from the consequences of the second claim.

In the opening statement to this symposium, Denning introduces the argument from artificiality, and discusses two main counterarguments: Science is possible in artificial domains; and natural forms of computing have been identified. However, two additional counterarguments are also worth mentioning. First, the distinction between natural and artificial is itself artificial and increasingly meaningless. Why are structures that are created by physical processes (such as planets and rivers) and various species of animal (such as anthills and beaver dams) considered natural, while comparable structures created by people are considered artificial? This appears to stem from the traditional notion that humans occupy a special status outside of nature, with their products therefore also being outside of nature. Yet we now understand people are as much a part of nature as any other biological organisms, and our products—including human-created forms of computation—are likewise just as much a part of nature. The distinction between natural and artificial phenomena, while still useful in

some contexts, is thus no longer defensible as a means of distinguishing science from non-science.

The second additional counterargument is our continually improving abilities to modify natural processes at finer levels of detail makes it increasingly difficult across the board to distinguish what is due to human agency and what is not. The only difference between natural and artificial flavorings is the feedstock; the molecules themselves are identical. Genetically modified plants are changed by human agency but are still plants. A liver grown from stem cells under laboratory-controlled conditions is a full biological organ. As with computing, much of the rest of science needs increasingly to be concerned with both natural and human-created (or human-modified) forms. Computing is thus becoming indistinguishable from the other sciences in terms of the nature of what it studies; and science as a whole is on a path to seek understanding of everything around and in us regardless of its genesis.

Each of the existing domains of science studies the interactions among a distinctive set of structures and processes. The physical sciences study physical (non-biological) structures—from atoms and molecules to rocks, planets and galaxies—plus the processes that operate on them. The life sciences study biological organisms and the processes of metabolism, growth, development, reproduction and aging that operate on them. The social sciences study the thought processes and behaviors of individuals and groups of people. As discussed in an earlier *Ubiquity* symposium, the computing sciences study information and its transformation. The field took an important step beginning in the 1990s when it was realized that this was in fact its proper subject of study rather than simply (human-made) computers. The earlier emphasis on computers may have blinded us to the existence of natural forms of computing. Now we can see various forms of computing are already embodied, for example, within living organisms.

Information can support representations and make distinctions. It is at the heart of the knowledge in our heads and of everything humans have written down on paper. It exists in many natural structures, as well as in the memories and data structures used by computers. Information is also at the core of communications, whether in modern computer networks or earlier oral, written, or wired traditions. The study of information has its roots in philosophy and mathematics, and it has always played an important role in the humanities. It has more recently led to the creation of such disciplines as information theory and information science. Many parts of computer science—from data structures and databases to Web technologies and artificial intelligence—also play a key role in the understanding of information. What truly distinguishes computer science from all of these other approaches though is its emphasis on not just information but also the dynamics of its transformation, as embodied in the diversity of computations enabled by combinations of hardware and software. This is why the experimental method is so important within computer science. Structures by themselves, whether informational or otherwise, often lend themselves to analytical forms of understanding. But when complex processes interact with structures, analytical methods typically fall short and experimental methods are the only means toward understanding.

If computing is science, where does it fit in the traditional taxonomy of the physical, life, and social sciences? Conceivably it could sit within one or more of the three existing domains; however, its core subject matter, of information and its transformation, simply does not fit within any of these domains. Instead, as laid out at some length in *On Computing*, it amounts to the fourth domain, and a full equal of the other three.

On Computing also introduces in some detail the "relational approach" to understanding the relationships between the core subject matter of computing and the cores of the other domains. Much of the book then analyzes how a pair of relationships among domains—implementation and interaction—can illuminate the scope and organization of computing as a scientific domain, while also highlighting the rich space of exciting multidisciplinary topics that are too often relegated to the fringes of the field. The analysis flows through four stages:

1. Monadic computing. Computing in isolation, as in theoretical computer science.
2. Pure dyadic computing. Computing relating to itself, as when a compiler implements one language in another and when multiple computers interact in a network.
3. Mixed dyadic computing. Computing relating to individual other domains, as in robotics and computational science.
4. Polyadic computing. Computing relating to multiple other domains, as in mixed reality and ubiquitous computing.

It is perhaps the multidisciplinary aspects of computing that have the most to offer STEM education, both as technologies for improving such education and as new topics of study in their own right. The potential for improving STEM education is clear and compelling, whether this involves simulation or virtual reality, intelligent tutoring systems or virtual humans, human-computer or brain-computer interfaces, or forms of augmented and mixed reality that combine the virtual with the real world. The potential as a subject for STEM education has received less attention overall; and is what will occupy the remainder of this essay. As a source of subject matter, multidisciplinary computing has the advantage over more traditional topics within computing of moving beyond the abstract space of pure information transformation to make contact with aspects of the world that are already familiar to students. Multidisciplinary computing is also the home of much of the current and future action in computing. Students find its topics exciting. Moreover, students who have worked with multidisciplinary computing should be incredibly well placed for the future, notably in terms of job opportunities. Let us look at a few examples, starting with the varieties of mixed dyadic computing—involving implementation or interaction with one other domain—and wrapping up with polyadic computing.

Consider first the implementation relationship. In one direction this yields implementations of computing, but not merely the familiar varieties of electronic hardware. Babbage's difference engine was, for example, a 19th century mechanical (i.e., physical) computer. Today students

can see implementations of computing across all three domains. There are chemical computers, biological computers, DNA computers, and social groups/networks that compute. There are even oddities such as billiard ball computers. Mechanical and social approaches to computing can be made concrete in the classroom in a highly visual and interactive manner; witness how Computer Science Unplugged engages student brains and bodies in implementing and understanding, for instance, a parallel sorting algorithm. Simulations of other approaches, such as DNA computing, could provide insight into both how the natural processes work—in this case how DNA replicates—and how alternative forms of computing operate, while introducing students to a multidisciplinary topic on the frontier of today's science and technology.

In the other implementation direction, we see computational simulation in general, with its many uses in science, engineering, education, training, communication and entertainment. At the core of most games sits a simulated environment of some sort—whether of the physical, life or social worlds, or some combination of all three. Simulations can provide an engaging means of studying STEM subjects, as exemplified in the previous paragraph, but they themselves can also be a useful and engaging topic for study within STEM education. And beyond simulation, true implementation of other domains by computing introduces mind-stretching scientific questions concerning whether computation underpins all of our physical reality, whether it is possible to create computational life, and whether artificial intelligence can yield human-level intelligence.

When we get to interaction, we see computational sensing and manipulation of the physical, biological and social worlds. This includes remote sensing and sensor networks; as well as robots of all sorts, from traditional industrial robots, to self-driving cars and unmanned aerial vehicles, to rapid prototyping devices that can build everything from toys to buildings. We also see an increasing variety of ways that computers interact with our bodies and minds. They understand what we say, interpret our movements, and provide rich, interactive sensory and informational experiences. They are now even starting to interface directly with our brains, providing the technological equivalent of mind reading. Inexpensive robots are already finding their way into STEM education, but this far from exhausts the space of possibilities. Consider just one additional example, studying the appropriate placement of a network of sensors that monitors a real physical environment, along with the integration of results across these sensors. This could combine lessons in mathematics and computing along with the study of whatever domain is being monitored.

Polyadic computing combines multiple domains and relationships to yield many of the most exciting and forward-looking topics under investigation within the computing sciences. It includes prosthetic devices, caretaker robots, intelligent robots, mixed reality environments that combine the virtual and physical worlds with human interaction, and ubiquitous computing environments that embed networked computing throughout the physical, and potentially also the biological, world. For STEM education, a device such as a prosthetic arm—likely in a

simplified form—could, for example, provide a wonderful opportunity for an integrated lesson spanning all four scientific domains.

In summary, computing is the basis for a true science, and in fact for an entire scientific domain. When analyzed via the relational approach, it can be seen to span a wide array of important and exciting topics that hold promise for engaging and educating our students, and for helping them to prepare for their, and our, future.

References

[1] Rosenbloom, P. S. A new framework for Computer Science and Engineering. *IEEE Computer* 37 (2004), 31-36.

[2] Denning, P. J. and Rosenbloom, P. S. Computing: The fourth great domain of science. *Communications of the ACM* 52, 9 (2009), 27-29.

[3] Rosenbloom, P. S. *On Computing: The Fourth Great Scientific Domain.* MIT Press, Cambridge, 2013.

About the Author

Paul S. Rosenbloom is a Professor of Computer Science at the University of Southern California and a Project Leader at USC's Institute for Creative Technologies. He received a BS degree in Mathematical Sciences (with distinction) from Stanford University and M.S. and Ph.D. degrees in computer science from Carnegie Mellon University. Before coming to USC he was a Research Computer Scientist at Carnegie Mellon University and an Assistant Professor of Computer Science and Psychology at Stanford University. At USC he spent twenty years at the Information Sciences Institute, including ten years leading new directions activities and a stint as Deputy Director. Prof. Rosenbloom's research focuses on cognitive architectures/systems that model the human mind while supporting the construction of artificially intelligent agents. He was a co-creator of the Soar architecture, and is developing a new architecture/system—Sigma—based on graphical models. Prof. Rosenbloom has also spent considerable time over the past decade exploring the nature, stature and role of computing as a scientific discipline, culminating in the book *On Computing: The Fourth Great Scientific Domain* (MIT Press, 2013).

DOI: 10.1145/2590528.2590530

