Ubiquity Symposium

What is Computation?

The Evolution of Computation

by Peter Wegner, Brown University

Computer science became an evolutionary discipline about 70 years ago, some 2,000 years after the Greeks created mathematics, physics and philosophy. However, the questions "What is Mathematics?" and "What is Physics?" remain unanswered, and it is not surprising that the specification of computation likewise remains unresolved, in spite of the increasing centrality of the computing discipline. Mathematics has evolved from Leibniz in the early 18th century, Gauss in the 19th century, Hilbert in 1900, and Gödel and Church in 1931. Physics has evolved from Newton to Einstein to quantum theory, and is currently enmeshed in string theory. Computing has evolved from Turing machines through object-oriented programming and the Internet to interactive and biological models. But the evolution of scientific disciplines does not determine their complete specification, which changes as new research proposals establish new modes of thought.

Janette Wing in CACM Jan 2008, raises five computational questions that remain unanswered: "Is P=NP?", "What is computable?", "What is Intelligence?", "What is Information?", and "How can we simplify complex systems?" This suggests that "What?" questions can be worthwhile and technically meaningful; formulating and examining questions is a useful endeavor that contributes to the technical evaluation of disciplines even if we are unable to explain them.

Turing's 1937 paper, "On Computable Numbers with an Application to the *Entscheidungs-problem*," strengthened Gödel's rejection of Hilbert's mathematical assertion that all theorems can be proved true or false by logical arguments, thereby expressing "What is Mathematics?" in terms of logical theorem proving. Moreover, Turing's interest in computation led him to show that computers could not solve all mathematical problems because the halting problem is unsolvable. The Church-Turing thesis asserted in 1938 that all effectively computable mathematical functions can be computed by Turing machine algorithms, restricting solvable problems to the category of mathematics. Turing machines were accepted by Gödel as the basic model of computation, and the Turing Award was established as the "Nobel Prize" for computer science. Turing award winners like Alan Perlis, Maurice Wilkes, and Richard Hamming were selected in part because their research contributions helped to address and understand the nature of models of Computation.

Turing was born in 1912, and his undergraduate work at Cambridge during 1931-1934 was primarily mathematical. Turing machines were judged as a mathematical interpretation of computational problem solving; and computing was interpreted as an entirely mathematical discipline. However, in the mid-1960s (ten years after Turing's death) the Church-Turing thesis was reinterpreted as a model that could solve all computable problems (well beyond mathematical functions); and Turing machines were seen as a complete problem-solving method. The original Church-Turing thesis was judged to imply the much stronger assumption that Turing machines could solve all computational problems and perform computational activities of all constructible computers. This strong Church-Turing thesis was adopted in the 1960s by the theory of computation community who believed that all computation was mathematical and that mathematics was the primary defining principle of scientific disciplines. But this led to a misleading interpretation of "What is Computing?" based on an inappropriate assumption about the nature of computation that had been refuted by Gödel and Turing in the 1930s and was discredited by computer experts like Robin Milner in the 1980s. My article with Dina Goldin, "Refuting the Strong Church-Turing Thesis" (Mind and Machines, March 2008) includes a detailed analysis of why the strong thesis is untrue.

Turing's substantial contributions to computation included the Enigma computer, which provided a method of locating and destroying German submarines during World War Two, and his paper "Computing Machinery and Intelligence" (1950), which proposed that machines were intelligent and could learn to think and act like humans. But Turing was unable to actually build Turing machines or other modern computers. My Cambridge mentor Maurice Wilkes, who constructed the first British computer (the EDSAC) in 1950, suggested that Turing was better at research than at building computers because his handling of collaborators was weaker than his control of ideas.

The expansion of computers has been considerable since the 1950s, with the growth of IBM, personal computers, and object-oriented programming and the creation of the Internet, Google, and concurrent programming systems, all of which go well beyond Turing machines in their computing and problem solving methods. One new concept, missing from early Turing machines, is "Interactive Computing," which accommodates interaction with the environment during the computation. Interaction occurs in driving cars, when we observe the impact of road conditions while driving home; and with one's spouse and children during a long marriage. Computation must be able to perform tasks that provide services to the environment or to society, rather than simply compute outputs from input specifications.

Together with my colleagues Dina Goldin and Scott Smolka, I edited a book, *Interactive Computation: The New Paradigm* (Springer Verlag 2007), whose eighteen articles on the role of interactive computing demonstrate that interaction is a new paradigm of computation. The first

chapter by Robin Milner, who passed away in March 2010, suggests that interaction is central to computing systems and that interactive computing requires the form of logic to be changed. His response to his 1991 Turing award (published by the CACM in 1993) is entitled "Elements of Interaction," showing that he clearly viewed interaction as a central element of computation.

Chapter 3, "Principles of Interactive Computation" (by Dina Goldin and myself), explores the role of Truth in human philosophical thinking, of rationalism versus empiricism as a model of problem solving, and of computational thinking as a form of intelligence.

Including human thought as a form of computation permits philosophical and religious problem solving to be included in computational reasoning. Rene Descartes believed truth was important but hard to establish, and proposed "I think therefore I am" (*cogito ergo sum*) as the only indubitably true statement and rationalist reasoning as the primary human mechanism for thought and action. Descartes' assumption served as a basis for the rationalist philosophy of Kant and Hegel and for European political models that led to disastrous restrictions on society based on incorrect interpretations of rationalist beliefs. Rationalism also led to questionable religious assumptions about the nature of God and of religious beliefs. The biblical book of Genesis represents God as forbidding Adam and Eve to eat of the tree of knowledge, and imposing a severe penalty for their disobedience by expelling them from the Garden of Eden. God apparently considered human acquisition of Truth more reprehensible than Cain's murder of Abel (described in the next chapter of Genesis). Fortunately, academicians are permitted to search for truth without being punished by society.

Rationalism holds that political actions are determined by the human mind, based on *a priori* insights about knowledge, whereas empiricism holds that knowledge is confirmed only by actual experience. Rationalism implies that people can achieve desired results through *a priori* intellectual assumptions, while empiricism implies that experiments are more effective than rationalism in determining Truth. Rationalism was widely adopted by European philosophers like Descartes and Kant and by political leaders of the French Revolution, Russian communism, and German fascism. Empiricism was sponsored by British philosophers like John Locke, who contributed to the evolution of Parliament, the founding of the Bank of England, the separation of Church and State in American democracy, and the growth of scientific research in terms of experimental (i.e., empirical) scientific research in the Cavendish Laboratory (with which I interacted while a graduate student at Cambridge).

The success of British empiricism in the evolution of the British Empire, compared with the relative failure of European rationalism in imposing undesirable socio-political laws, suggests that empiricism may provide a better practical basis than rationalism for human progress. Interaction is an empirical as opposed to rationalist model of computation, which Turing

developed in his 1937 paper, and which is described in detail in my recent edited book, *Interactive Computation: The New Paradigm*. The strong Church-Turing thesis may be viewed as a rationalist transformation of Turing's empirical model, which failed in part because computer problem solving cannot be determined by rationalist mental processes but must be empirically tested.

Turing's 1950s paper on "Machinery and Intelligence" suggested that machines could think by responding to questions as humans do, and that Turing machines would be able to play chess and perform other intelligent tasks by the end of the century. Skeptics believed that machines were inherently unable to think in this way because such behavior does not capture human awareness by merely simulating human behavior, and that some intelligent tasks cannot be performed in this manner because computer intelligence is inherently weaker than that of humans. In particular, Penrose claimed that physics could not be completely modeled by Turing machines through algorithms. We agree with Penrose in principle, but suggest that interactive computers can completely model Turing's form of human thinking because their modeling ability is greater than that of Turing machines. Interaction machines can perform more powerful forms of computing than Turing machines, and can perform the kind of thinking proposed by Turing because interaction improves their performance over that of Turing machines.

The evolution of curricula for teaching computing is still unresolved, with continuing conflict between theoretical (mathematical) and practical (engineering) forms of instruction. I was personally involved in the publication of Curriculum 68 (CACM 1968), the first major description of the curriculum soon after computer science had become a recognized undergraduate discipline. Peter Denning sponsored a debate on teaching computer science in 1989, with responses by Parnas, Hamming, Karp and others to an article by Edsgar Dijkstra on "The Cruelty of Teaching Computer Science." The current debate on "What is Computation?" may be viewed as a further response by specialists to the evolution of computing at a time when neither the content nor the curriculum is well specified.

Brown University's recent Von Neumann symposium (May 3-7, 2010) reviewed Von Neumann's contributions to mathematics, physics, computation, and biology, with speakers John Conway (mathematics), Leon Cooper (physics), Richard Karp (computation), and Eric Davidson (biology), as well as Marina Von Neumann Whitman (von Neumann's daughter). Von Neumann's last book, *The Computer and the Brain* (published posthumously in 1958) was discussed in the context of current models of the Brain. It is unfortunate that no current scientist as broadly based as Von Neumann appears to exist, perhaps the breadth of scientific topics is more difficult to understand than in the past.

I will conclude by reporting briefly on some books that I believe have contributed to my understanding of computer science and other scientific disciplines. Fred Brooks' *The Mythical Man-Month* (1975) focuses on the reasons why the length of time needed for constructing software is so difficult to predict. His view that there is no "silver bullet" corresponds to our assertion that there is no definitive way of expressing "What is Computation?" Thomas Kuhn's *The Structure of Scientific Revolutions* (1962) suggests that new scientific paradigms are often rejected by believers in old paradigms precisely because they destroy old forms of reasoning and propose new forms of research. Herbert Simon's *The Sciences of the Artificial* (1970) suggests that computer science is an artificial rather than a purely natural science, but that artificial sciences are just as fruitful as natural ones. Stephen Hawking's *Cambridge Lectures* (1996), written while he occupied Newton's former chair as Lucasian professor at Trinity College, explores a new model of physics and especially the role of time, origins, black holes, computation and quantum theory in understanding Newton's, Einstein's, and Heisenberg's differing interpretations of physics, John Horgan's *The End of Science* (1997) interviews numerous well-known scientists about their research views and concludes that many of them believe scientific research has come to an end and is unlikely to develop significant new models as fruitful as the old models. I personally disagree with this; but Horgan's review of the opinions of well-known scientists is worth reading. A new book, *Church's Thesis After 70 Years*, edited by Olszewski, Woleński, and Janusz (Ontos, 2006) presents a variety of articles about the Church-Turing thesis, including its impact and its limitations in describing new models such as the Internet and concurrency associated with the evolution of computers. Bertrand Russell's *History of Western Philosophy* (1948) is one of the better books on the history of philosophy, which includes presentations of classical Greek, religious, Renaissance, and modern philosophies by one of the greatest philosopher/scientists of the 20th century, whose life spanned almost 100 years (1872-1970). Last but not least, my book, *Programming Languages, Information Structures, and Machine Organization* (PLISMO) (McGraw Hill 1969), is one of the earliest attempts to provide a substantial explanation of the role of programming languages and information structures as models of computing (explanations of "What is Computation?").

Computational specification and problem solving has undergone a paradigm shift from mathematical mainframes to workstations, object-oriented programming, and the Internet which has expanded Turing machine models to interactive systems whose physical and problem solving form cannot be completely specified, but certainly goes well beyond Turing machines, algorithms and mathematical specification to new models of computation that will change as science and society continue to evolve.

About the Author

Peter Wegner (pw@cs.brown.edu) is an emeritus professor at Brown University and was inducted as a Fellow of the ACM in 1995. Wegner is also the former editor-in-chief of *ACM Computing Surveys*.

DOI: 10.1145/1880066.1883611

