Computation Beyond Turing Machines

Peter Wegner, Brown University   *Dina Goldin, U. of Connecticut*

1. Turing's legacy

Alan Turing was a brilliant mathematician who showed that computers could not completely prove mathematical assertions, extending Gödel's proof that logic could not completely model mathematical truth. This connection between computers and mathematics was later used to develop a mathematical foundation for computer science, comparable to mathematical foundations for physics and other sciences. The present article shows that Turing machines are inappropriate as a universal foundation for computational problem-solving, and that computer science is a fundamentally non-mathematical discipline. Though interaction is not the only way to extend computation beyond Turing machines, we show that Turing, Milner, and others have used interaction for this purpose.

Turing was born in 1912. He was accepted by Cambridge in 1930 to study mathematics, and became a Fellow of Kings College in 1934 at the age of 22, completing a dissertation that extended group theory models of Von Neumann. His 1936 paper, "On Computable Numbers with an application to the *Entscheidungsproblem*", proved that mathematics could not be completely modeled by computers. In the early 1940s he developed a computer model of German cipher code that helped England win World War II. In the late 1940s he developed computational models of artificial intelligence, chess, and the human mind, suggesting that computers could completely model human thought and would play chess better than humans before the end of the century. Though the 1936 paper was primarily about the inability of Turing machines to solve mathematical problems, Turing machines were adopted by theoretical computer scientists in the 1960s as a mode of solving all problems of computing. In this essay we examine the historical evolution of Turing's model from mathematical weakness in the 1930s to computational strength in the 1960s, and then to computational weakness in the 1990s as increases in the applicability of computation broadened our notion of "computational problem" and revealed the limitations of the power of Turing machines to handle problem solving.

2. Hilbert, Gödel, and Church

In 1900 Hilbert proposed that logic could completely prove the truth or falsity of mathematical assertions, and listed 25 unproven mathematical assertions that mathematicians should try to prove. Russell and Whitehead's *Principia Mathematica* accepted Hilbert's principle and provided an account of

mathematical logic as a universal model of mathematical provability. The failure to achieve their goals led to Gödel's 1931 proof that logic could not prove all mathematical theorems [Go31]. Gödel showed that the *Entscheidungsproblem* ("decision problem") was in principle unsolvable by logic, and this led to work by many mathematicians to further explain the theory and philosophy of mathematical unsolvability in terms of logic or other models of mathematics.

Gödel's ideas were taken up by Church, who proved in 1935 that the *Entscheidungsproblem* could not be solved by the lambda calculus. By contrast, Turing showed by computers that the *Entscheidungsproblem* could not be solved, because the "halting problem" of Turing machines was itself unsolvable. Turing's result was accepted by Gödel and Church as a simpler and better unsolvability argument. Turing was invited to Princeton in 1937 to work with Church on what was later called the Church-Turing thesis. This thesis equated logic, lambda calculus, Turing machines, and algorithmic computing as equivalent mechanisms of problem solving. This thesis was later reinterpreted as a uniform complete mechanism for solving all computational problems.

Turing implied in his 1936 paper that Turing machines (which he called *automatic machines*, or *a-machines*) could not provide a complete model for all forms of computation, just as they could not provide a model for all forms of mathematics. He defined *c-machines* (choice machines) as an alternative model of computation, which added interactive choice as a form of computation; later, he also defined *u-machines* (unorganized machines) as another alternative that modeled the brain. But Turing did not formalize them, and they were discarded in the 1960s as unnecessary, because it was assumed that Turing machine models could completely describe all forms of computation. Though this narrow interpretation of the Church-Turing thesis contradicted Turing's assertion that Turing machines could only formalize *algorithmic* problem-solving, it was accepted by the CS community in the 1960s and became a dogmatic principle of the theory of computation. Computer science was modeled by a mathematical model of theoretical Turing machines whose scientific model paralleled those of physics, chemistry, and biology, providing an acceptable but weak theory of computation.

3. From algorithms to interaction

After its beginning with Turing's pioneering paper in 1936, computer science emerged as a mature discipline in the 1960s, when universities nationwide started offering it as an undergraduate program of study. By 1968, there was a general consensus on what should be taught as part of this new discipline, enunciated in ACM's Curriculum '68 document [ACM68]. The new discipline of computer science viewed computation as *information processing*, a *transformation* of *input* to *output* -- where the input is completely defined before the start of computation, and the output provides a solution to the problem at hand. Such mechanistic transformations were long known in mathematics as *algorithms*; the approach to

computation adopted by computer science is hence referred to as *algorithmic.*

The field of computing has greatly expanded since the 1960s, and it has been increasingly recognized that artificial intelligence, graphics and the Internet cannot be expressed by Turing machines. In each case, *interaction* between the program and the world (*environment*) that takes place during the computation plays a key role that cannot be replaced by any set of inputs determined prior to the computation. In the case of artificial intelligence, interaction can be viewed as a prerequisite for intelligent system behavior, as argued by Brooks [Br91]:

> "Real computational systems are not rational agents that take inputs, compute logically, and produce outputs… It is hard to draw the line at what is intelligence and what is environmental interaction. In a sense, it does not really matter which is which, as all intelligent systems must be situated in some world or other if they are to be useful entities".

British computer scientist Robin Milner developed a new conceptual framework for models of computation, based on CCS and later the ?-calculus. In his Turing Award lecture "Elements of Interaction" [Mi93], Milner asserts that established models of computation are insufficient:

> "Through the seventies, I became convinced that a theory of concurrency and interaction requires a new conceptual framework, not just a refinement of what we find natural for sequential [algorithmic] computing".

4. Beyond Turing machines

Milner's Turing Award lecture in 1991 presented models of interaction as complementary to the closed-box computation of Turing machines. However, he avoided the question whether the computation of CCS and the ?-calculus went beyond Turing machines and algorithms. TMs had been accepted as a principal paradigm of complete computation, and it was premature to openly challenge this view in the late '70s and the early '80s. In the last two decades, computing technology has shifted from mainframes and microstations to networks and wireless devices, with the corresponding shift in applications from number crunching and data processing to embedded systems and graphical user interfaces. We believe it is no longer premature to encompass interaction as part of computation. A paradigm shift is necessary in our notion of computational problem solving, so it can provide a complete model for the services of today's computing systems and software agents.

The model of interaction machines as an extension of Turing machines was developed in the late

1990s [Weg97]. The theoretical framework has been improved in [GSW01]. Van Leeuwen, a Dutch expert on the theory of computation, wrote an article extending computers beyond Turing machines [VLW00], which referred to these recent models of interaction, admitting that

> "the classical Turing paradigm may no longer be fully appropriate to capture all features of present-day computing."

Our concept of interactive models was questioned because we originally failed to provide a theoretical framework comparable to that for Turing Machines. However, complete models of computation have often been developed with no theoretical foundation or mathematical models. Even Turing presented *c-machines* [Tu36] and *u-machines* [Tu48] without a formal foundation.

Though mathematics was adopted as a goal for modeling computers in the 1960s by analogy with models of physics, Gödel had shown in 1931 that logic cannot model mathematics [Go31] and Turing showed that neither logic nor algorithms can completely model computing and human thought. In addition to interaction, other ways to extend computation beyond Turing machines have been considered, such as computing with real numbers. However, the assumption that all of computation can be algorithmically specified is still widely accepted. Interaction machines have been criticized as an unnecessary Kuhnian paradigm shift. But Gödel, Church, Turing, and more recently Milner, Wegner and Van Leeuwen have argued that this is not the case.

Bibliography

[ACM68] Association for Computing Machinery, "Curriculum '68: Recommendations for academic programs in computer science", In *ACM Curricula recommendations for computer science,* Association for Computing Machinery, 1968.

[Br91] Rodney A. Brooks, "Intelligence Without Reason", MIT AI Lab Technical Report #1293.

[Go31] Kurt Gödel, "On Formally Undecidable Propositions of Principia Mathematica and Related Systems", Monatshefte fur Mathematik und Physik (38), 1931 (in German); English translation in *The Undecidable*, ed. Martin Davis, Raven Press 1965.

[GSW01] Dina Goldin, Scott Smolka, Peter Wegner, "Turing Machines, Transition Systems, and Interaction", *8th Int'l Workshop on Expressiveness in Concurrency*, Aarlborg, Denmark, August 2001.

[Mi93] Robin Milner, "Elements of Interaction" (Turing Award lecture), *Communications of the ACM* (36:1), 1993.

[Tu36] Alan Turing, "On Computable Numbers with an Application to the *Entscheidungsproblem*", *Proceedings of the London Math Society* 2 (42), 1936, pp. 173-198.

[Tu48] Alan Turing, "Intelligent machinery", in D. C. Ince, *Mechanical intelligence*, North-Holland, 1992, pp. 107-127.

[VLW00] Jan van Leeuwen, Jiri Wiedermann, "The Turing Machine Paradigm in Contemporary Computing", in *Mathematics Unlimited - 2001 and Beyond*, eds. B. Enquist and W. Schmidt, LNCS, Springer-Verlag, 2000.

[Weg97] Peter Wegner, "Why Interaction is More Powerful than Algorithms*, Communications of the ACM* 40 (5), 1997.