Principles of Interactive Computation

Dina Goldin and Peter Wegner

Brown University, Providence, RI, USA

Summary. This chapter explores the authors' 10-year contributions to interactive computing, with special emphasis on the philosophical question of how truth has been used and misused in computing and other disciplines. We explore the role of *rationalism* and *empiricism* in formulating true principles of computer science, politics, and religion. We show that interaction is an empiricist rather than rationalist principle, and that rationalist proponents of computing have been the strongest opponents of our belief that interaction provides an empirical foundation for both computer problem solving and human behavior. The rationalist position was adopted by Pythagoras, Descartes, Kant, and many modern philosophers; our interactive approach to computing suggests that empiricism provides a better framework for understanding principles of computing.

We provide an empirical analysis of questions like "can machines think", and "why interaction is more powerful than algorithms". We discuss *persistent Turing machines* as a model of sequential interaction that formally proves the greater power of interaction over algorithms and Turing machines. We explain that the *Strong Church–Turing Thesis*, formulated by theorists in the 1960s, violates Turing's original thesis about unsolvability of the decision problem and is a myth, in the sense that it departs from the principles of Turing's unsolvability result in his 1936 paper. Our analysis contributes to the book's goals towards the acceptance of interactive computing as a principle that goes beyond Turing machine models of computer problem solving.

1 Scientific, Political, and Religious Truth

Alan Turing's 1936 paper "On computable numbers with an application to the *Entscheidungsproblem* (decision problem)" [12] played a central role in the 1960s in establishing a mathematical paradigm of computation. Turing's goal was to show that Hilbert's decision problem was unsolvable in the sense that computers could not prove the truth or falsity of mathematical theorems. His paper strengthened Godel's earlier proof that mathematical theorems were not provable by logic, and weakened the belief in strong mental mathematical

ability, showing that human mathematical theorem proving through logic or computing was mentally incomplete.

However, such weakness in modeling mathematics was unwelcome to mathematical thinkers who believed that human reasoning could completely express mathematical ideas about the world. They believed that mathematics was a widespread scientific method for reasoning about physics and computation and that human thought provided a basis for scientific, philosophical, political, and religious understanding. They reinterpreted Turing's paper proving that computers could not solve all mathematical theorems, wrongly asserting that computers could in fact solve all computable problems (including mathematical problems), and that all computation could be done by Turing machines through algorithmic solution methods. Though Turing clearly showed this to be untrue, the desire to believe that computers and rationalist humans could solve a complete range of problems was so strong that Turing's counterarguments could easily be brushed aside and ignored.

There are many applications where humans consider it more important to adopt and justify principles rather than to prove them true. This is so in politics, where politicians have tenaciously preserved dubious principles in order to consolidate their power, regardless of whether the principles are true or ethical. This occurred in Germany under Hitler's Nazi principles, which he retained as a justification for dominating Germany and Europe until he was defeated in a costly war. It occurred under Stalin, who used Communist principles to eliminate his adversaries until he was himself eliminated, and more recently under Saddam Hussein and other democratically elected dictators. It has led to a decline of European scientific principles about the world in favor of extraneous political ideas.

Religions also seek to retain strongly established a priori beliefs independently of their truth. Christianity, Judaism, and Islam preserve their belief in God and in the validity of biblical texts that distinguish their religion from other religions, and can eliminate and kill nonbelievers simply because their beliefs differ, independently of their truth. Truth is adjusted so that religious belief is inherently true and is used to destroy alternative ideas about society independently of the truth or falsity of religious or secular ideas. For example it is appropriate to discredit Darwin's evolution theory because it negates the biblical account of creation in spite of its experimental validity, just as Copernican and Galilean models were discredited three hundred years earlier.

The questionable manipulation of truth in politics and religion is widely acknowledged, but is nevertheless accepted and practiced by particular political and religious organizations. Scientists have assumed that truth is more often falsified by philosophical experts than by scientific researchers, but careful analysis shows that this is not always the case and that truth claims among scientists like Newton and Einstein, or mathematicians like Hilbert, can be as false as the truth claims of political and religious experts. Newtonian physics was assumed indubitably true for 200 years until modified by Einstein's theory of relativity, while Descartes philosophical assumption that "Cogito Ergo

Sum" is indubitably true is seen in retrospect as a questionable assumption that has been used to support many untrue beliefs on the basis of rationalist principles that can be easily disproved by empiricism.

2 Rationalism Versus Empiricism

Rationalism holds that truth is determined by the human mind in terms of "a priori" (predetermined) insight about knowledge, while empiricism holds that knowledge is confirmed only by experience of actual perceptions that determine knowledge. Rationalism implies that people can strongly advocate scientific, political, or religious knowledge through "a priori" mental properties of the brain that are inherently true and cannot be changed by experiments, while empiricism implies that experiments are more effective than predetermined a priori properties of the brain in determining scientific, political, or religious knowledge. Since rationalists believe humans have smarter forms of understanding than do empiricists, and can ignore empirical forms of knowledge, rationalism is often adopted as a broader and more complete form of knowledge, even though it can support wrong and sometimes disastrous principles.

The adoption of rationalism by Pythagoras as an a priori basis for mathematical truth led to its adoption by Plato, who focused on geometry as a central rationalist discipline whose a priori truth implied that a priori principles were a central justification of human knowledge. Aristotle accepted Plato's rationalist view of truth, though his idea of the syllogism was in part empiricist (Socrates mortality was due to the empirical fact that all men are mortal). Though some scientists and philosophers accepted empiricism, the much greater practical power of rationalism helped to establish its role as a primary basis for knowledge about the world and society. This was strengthened by the choice of rationalism as a primary basis for religious beliefs like the existence of God, and the truth of biblical narrative (which could not be proved by empiricism though easily acceptable through rationalism).

St. Augustine (fifth century) and St. Thomas Aquinas (thirteenth century) developed rationalist philosophical models of religion that redefined Christian beliefs in ways that are still accepted today. Descartes is considered the world's greatest modern philosopher primarily because his Jesuit upbringing allowed him to define philosophy in terms of rationalist religious principles at a time when it was being questioned both by scientists like Galileo and by religious dissenters like Martin Luther. Newton solidified scientific principles of Galileo, but spent the last 30 years of his life studying religion. Detailed analysis of philosophers like Descartes and Kant makes it clear that the basis for acceptance of philosophical ideas had more to do with their contributions to religious thought than with their inherent truth or the strength of their arguments.

Locke, Berkeley, and Hume are among the few widely studied empiricist philosophers who contributed substantially to human and political thought. All three were strongly challenged by rationalist opponents, but contributed to the strength of British and US politics though not to European politics. Locke had to flee to Holland during the short Catholic reign of James II (1685–88) to avoid imprisonment and potential death in the Tower of London as a Protestant dissenter. His ideas contributed to the power of the British Parliament, to the Bank of England, and to the US Constitution. His essay on religious toleration, written while in exile in Holland to support toleration between Protestants and Catholics, was used in the US Constitution to support separation of church and state. Locke's contributions to both the growth and power of the British empire and the rise of US democracy suggests that empiricism properly applied can contribute to both the quality and the persistence of political democracies.

Though empiricism has enhanced both scientific research and political democracy, it could not displace rationalism in European politics or in widespread religious beliefs. Kant's early work was influenced by Hume's empiricism, but his later written *Critique of Pure Reason* was strongly rationalist, advocating a priori knowledge over experiment as a basis for acceptance of reason and truth. Kant's model led to the rationalist philosophy of Hegel, which in turn influenced the communist rationalism of Marx and the Nazi rationalism of Hitler. Contemporary politicians like US president Bush are strongly rationalist, using a priori political and religious certainty to support principles like the war in Iraq or the sanctity of marriage in contradicting empiricist assertions about human nature raised by their opponents.

Mathematicians have traditionally believed that mathematics is justified by rationalist rather than empiricist principles because properties of numbers, geometry, and equations are a priori and therefore rational. Hilbert's assumption that all mathematical assertions could be logically proved was considered an a priori idea, and its empirical disproof by Godel and Turing was considered suspect because empiricism should not intrude on a priori inherently rationalist principles. Turing's proof that computers could not automatically decide all mathematical theorems was likewise an empiricist disproof of an a priori rationalist idea, and the fast and loose idea that Turing machines can solve all computable problems was a return from empiricist to previously accepted rationalist a priori results.

The choice of interaction as a computational extension of Turing machines can be viewed as an empiricist model of computing associated with Turing's original empiricist assertion. The strong resistance to this view is in part due to the idea that empiricist models should not intrude on a priori rationalist assumptions about the nature of computation. It is for this reason that we have begun this chapter with a philosophical discussion of the role of empiricism and rationalism in processes of computation and human thought.

3 Turing's 90th Birthday

Turing was born in 1912 and died tragically in 1954 around his 42nd birthday, committing suicide because he was being prosecuted by the police as a homosexual. His 90th birthday conference in Lausanne in 2002 yielded a book about his life and legacy [11] with articles by Andrew Hodges, Martin Davis, Daniel Dennett, Jack Kopeland, Ray Kurzweil, and many other writers including the editor Christof Teuscher and the authors of this chapter.

Andrew Hodges, author of a comprehensive book on Turing, reviews his life and examines what Turing might have contributed had he lived longer. Copeland explores Turing's contributions to artificial intelligence, artificial life, and the Turing Test of whether machines can think. Teuscher explores his contributions to neural networks and unorganized machines. The authors show that Turing's contributions are much broader than Turing machines, and include interaction as a super-Turing model that Turing had already examined through choice machines, oracles, and unorganized machines.

Several writers used this opportunity to explore the pros and cons of hypercomputation as an extension of Turing machines. Martin Davis claimed that hypercomputation simply shows that noncomputable inputs may yield noncomputable outputs and that all computable problems can in fact be solved by Turing machines. We show that algorithms can express only a subset of computable problems and that interaction provides a framework for expressing non-algorithmic problems and extending Turing machine models.

Turing machines and algorithms must completely specify all inputs before they start computing, while *interaction machines* [17] can add actions occurring during the course of the computation. Driving home from work is an example of a computation where actions observed during the course of driving must be included in deciding how to drive and is therefore an example of an interactive non-algorithmic computation. Drivers must observe the road conditions, the cars in front of them, the traffic lights, and pedestrians crossing the street in order to decide how to drive and whether to change the speed or the direction of driving. This eliminates a predefined algorithmic specification of exactly how and where to drive and shows that interaction is more expressive than algorithms in the context of driving home.

Other similar extensions of interactive over algorithmic specification include operating systems, managing a company, fighting opponents in a war, or even aiding one's partner in a marriage. Interactive computations are more powerful than algorithmic computations of Turing machines in many practical situations that occur frequently in computing. Their power does not depend on the quality of prior inputs as suggested by Martin Davis, but it does depend on the degree to which the environment can be observed and acted upon during the course of the computation.

4 Can Machines Think?

Turing in his 1950 paper "Machinery and intelligence" [14] suggests that intelligence should be defined by the ability of machines to respond to questions exactly like humans, so that their ability to think and understand cannot be distinguished from that of humans. Turing not unexpectedly equated "machines" with "Turing machines". He permitted machines to delay their answer to mimic the slower response time of humans in games or mathematical computing, but did not consider that machines can sometimes be inherently slower than humans, or require hidden interfaces from agents or oracles when they answer questions.

Skeptics who believe that machines cannot think can be divided into two classes:

- *intentional skeptics* who believe that that machines that simulate thinking cannot think, because their behavior does not completely capture inner (intentional) awareness or understanding;
- *extensional skeptics* who believe that machines have inherently weaker extensional behavior than humans, because they cannot completely model physics or consciousness.

Searle is an intentional skeptic who argues that passing the test intentionally did not constitute thinking because competence did not constitute inner understanding, while Penrose [7] asserts that machines are not extensionally as expressive as physical or human mental models.

We agree with Penrose that Turing machines cannot model the real world, but disagree that this implies extensional skepticism because interaction machines can model physical behavior of the real world and mental behavior of the brain. Our assertion that interaction is more powerful than algorithms implies not only greater computing power but also greater thinking power of interactive machines.

Penrose builds an elaborate house of cards on the noncomputability of physics by Turing machines. However, this house of cards collapses if we accept that Turing machines do not model all of computation. Penrose's argument that physical systems are subject to elusive noncomputable laws yet to be discovered is wrong, since interaction is sufficiently expressive to describe physical phenomena like action at a distance, nondeterminism, and chaos, which Penrose cites as examples of physical behavior not expressible by computers. Penrose's error in equating Turing machines with the intuitive notion of computing is similar to Plato's identification of reflections on the walls of a cave with the intuitive richness of the real world. Penrose is s self-described Platonic rationalist whose arguments based on the acceptance of Church's thesis are disguised forms of rationalism, denying first-class status to empirical models of interactive computation.

Penrose's dichotomy between computing on the one hand and physics and cognition on the other is based on a misconception concerning the nature of

computing that was shared by the theorists of the 1960s and has its roots in the rationalism of Plato and Descartes. The insight that the rationalist/empiricist dichotomy corresponds to algorithms and interaction and that "machines" can model physics and cognition through interaction, allows computing to be classified as empirical along with physics and cognition. By identifying interaction as an ingredient that distinguishes empiricism from rationalism and showing that interaction machines express empirical computer science, we can show that the arguments of Plato, Penrose, and rationalist computer scientists of the 1960s are rooted in a common fallacy concerning the role of noninteractive algorithmic abstractions in modeling computation in the real world.

5 Why Interaction is More Powerful than Algorithms

The paper by this title [16] was a primary early attempt to explore the distinction between algorithms and interaction. It was widely praised by practical programmers but criticized by mathematical rationalists who believed that Turing machines express all forms of problem solving and computation. However, algorithms yield outputs completely expressible by memoriless, history-independent inputs, while interactive systems like personal computers, airline reservation systems, and robots provide history-dependent services over time that can learn from and adapt to experience.

Algorithms are "sales contracts" that deliver outputs in exchange for an input, while interactive system specifications are "marriage contracts" that specify their behavior for all contingencies (in sickness and in health) over the lifetime of the object (till death do us part). The folk wisdom that marriage contracts cannot be reduced to sales contracts is made precise by showing that interaction cannot be reduced to algorithms.

Interaction provides a better model than Turing machines for object-oriented programming. Objects are interactive agents that can remember their past and provide time-varying services to their clients not expressible by algorithms. It is fashionable to say that everyone talks about object-oriented programming but no one knows what it is. But knowing what it is has proved elusive because of the implicit assumption that explanations must specify what it is by algorithms, that excludes specifyng what it is through interaction. The better explanation of computational behaviors through interaction is similar to that used in better expressing the notion "can machines think", and occurs also in many other descriptions of computing.

Interactive extensions of Turing machines through dynamic external environments can be called interaction machines. Interaction machines may have single or multiple input streams, synchronous or asynchronous actions, and can differ along many other dimensions. Interaction machines transform closed to open systems and express behavior beyond that computable by algorithms in the following ways:

Claim: Interaction machine behavior is not expressible by Turing machine behavior.

Informal evidence of richer behavior: Turing machines cannot handle the passage of time or interactive events that occur during computation.

Formal evidence of irreducibility: Input streams of interaction machines are not expressible by finite inputs, since any finite representation can be dynamically extended by uncontrollable adversaries.

The radical view that Turing machines are not the most powerful computing mechanism has a distinguished pedigree. It was accepted by Turing who assumed in 1936 that choice machines were not expressible by Turing machines and showed in 1939 that oracles for predicting noncomputable functions were not Turing machines. Milner noticed as early as 1975 that concurrent processes cannot be expressed as algorithms, while Manna and Pnueli showed in 1980 that nonterminating reactive processes like operating systems cannot be modeled by algorithms.

Input and output actions of processes and objects are performed with logical sensors and effectors that change external data. Objects and robots have very similar interactive models of computation: robots differ from objects only in that their sensors and effectors have physical rather than logical effects. Interaction machines can model objects, software engineering applications, robots, intelligent agents, distributed systems, and networks like the Internet and the World-Wide Web.

6 Theory of Sequential Interaction

The hypothesis that interactive computing agents are more expressive than algorithms requires fundamental assumptions about models of computation to be reexamined. What are the minimal extensions necessary to Turing machines to capture the salient aspects of interactive computing? This question serves as a motivation for a new model of computation called *persistent Turing machines* (PTMs), introduced by Goldin et al. [3]; van Leeuwen and Wiedermann's chapter in this book provides a related model, with similar motivations [15]. PTMs allow us to formally prove Wegner's hypothesis regarding the greater expressiveness of interaction.

PTMs are interaction machines that extend Turing machine semantics in two different ways, with dynamic streams and persistence, capturing sequential interactive computations. A PTM is a nondeterministic three-tape Turing machine (N3TM) with a read-only input tape, a read/write work tape, and a write-only output tape. Its input is a *stream of tokens* (strings) that are generated *dynamically* by the PTM's environment during the computation.

A PTM computation is an infinite sequence of *macrosteps*; the i'th macrostep consumes the i'th input token a_i from the input stream, and produces the i'th output token for the output stream. Each macrostep is an

N3TM computation consisting of multiple N3TM transitions (microsteps), just as each input and output token is a string consisting of multiple characters. The input and output tokens are temporally interleaved, resulting in the interaction stream $\{(a_1, o_1), (a_2, o_2), ...\}$. This stream represents the *observed behavior* of the PTM during the computation.

PTM computations are *persistent* in the sense that a notion of "memory" (work-tape contents) is maintained from one macrostep to the next. Thus the output of each macrostep o_i depends both on the input a_i and on the work tape contents at the beginning of the macrostep. However, the contents of the worktape is hidden internally, and is not considered observable. Thus this contents is not part of interaction streams, which only reflect input and output (observable) values.

Persistence extends the effect of inputs. An input token affects the computation of its corresponding macrostep, including the work tape. The work tape in turn affects subsequent computation steps. If the work tape were erased, then the input token could not affect subsequent macrosteps, but only "its own" macrostep. With persistence, a macrostep can be affected by all preceeding input tokens; this property is known as *history dependence*.

Three results concerning the expressiveness of PTMs are discussed below. The first result is that the class of PTMs is isomorphic to *interactive transition systems* (ITSs), which are effective transition systems whose actions consist of input/output pairs, thereby allowing one to view PTMs as ITSs "in disguise". This result addresses an open question concerning the relative expressive power of Turing machines and transition systems. It has been known that transition systems are capable of simulating Turing machines. The other direction, namely "What extensions are required of Turing machines so they can simulate transitions systems?", is solved by PTMs.

The second result is the greater expressiveness of PTMs over *amnesic Turing machines* (ATMs), which are a subclass of PTMs that do not have persistence, in effect by erasing their work tape. ATMs extend Turing machines with dynamic streams but without memory. An example is a squaring machine, whose input and output are streams of numbers; at i'th macrostep, if the input number is a_i, the output is its square a_i^2. While some have found it tempting to think that only dynamic streams are needed to model interaction, such as [9], our results show that persistence (memory) is also necessary. Furthermore, since ATMS are an extension of Turing machines, the strictly greater expressiveness of PTMs over ATMs also implies that PTMs are more expressive than Turing machines.

The third result proves the existence of a *universal* PTM; similarly to a universal Turing machine, a universal PTM can simulate the behavior of any arbitrary PTM.

PTMs perform sequential interactive computations, defined as follows:

Sequential Interactive Computation: A sequential interactive computation continuously interacts with its environment by alternately accepting an

input string and computing a corresponding output string. Each output-string computation may be both nondeterministic and history-dependent, with the resultant output string depending not only on the current input string, but also on all previous input strings.

PTMs do not capture all forms of interactive computation. Interaction encompasses nonsequential computation as well, specifically multistream, or multiagent, computation [17]. However, examples of sequential interactive computation abound, including Java objects, static C routines, single-user databases, and network protocols. A "simulator PTM" can be constructed for each of these examples, similarly to the construction of the universal PTM. The result is a sequential interactive analogue to the Church–Turing thesis, stating that PTMs capture all sequential interaction:

Sequential Interaction Thesis: Any sequential interactive computation can be performed by a persistent Turing machine.

This hypothesis establishes the foundation of the theory of sequential interaction, with PTMs and ITSs as its alternative canonical models of computation. Since PTMs are more expressive than amnesic TMs and Turing machines, this theory represents a more powerful problem-solving paradigm than the traditional theory of computation (TOC), confirming the conjecture that "interaction is more powerful than algorithms". We also expect that this theory will prove as robust as TOC, with appropriate analogues to fundamental TOC concepts such as logic and complexity.

7 The Church–Turing Thesis Myth

The greater expressiveness of interaction over Turing machines is often viewed as violating the *Church–Turing thesis* (CTT). This is a misconception, due to the fact that the Church–Turing thesis has been commonly reinterpreted; we call this reinterpretation the *Strong Church–Turing thesis* (SCT). In this section, we show that the equivalence of the two theses is a myth; a longer discussion can be found in [4]. Our work disproves SCT, without challenging the original Church–Turing thesis.

The Church–Turing thesis, developed when Turing visited Church in Princeton in 1937–38 and included in the opening section of [13], asserted that Turing machines and the lambda calculus could compute all algorithms for effectively computable, recursive, mathematical functions.

Church–Turing thesis (CTT): Whenever there is an effective algorithm for computing a mathematical function it can be computed by a Turing machine or by the lambda calculus.

While *effectiveness* was a common notion among mathematicians and logicians of early twentieth century, it lacked a formal definition. By identifying

the notion of effective function computability with the computation of Turing machines (as well as the lambda calculus and recursive functions), the Church–Turing thesis serves to provide a formal definition in the case of effective computation of functions, based on transformations of inputs to outputs. However this thesis was extended in the 1960s to a broader notion of computability, which we call the Strong Church–Turing thesis.

Strong Church–Turing thesis (SCT): A Turing machine can compute anything that any computer can compute. It can solve all problems that are expressible as computations (well beyond computable functions).

While the Church–Turing thesis is correct, this later version is not equivalent to it; in fact, PTMs prove it wrong. Since they are inequivalent, a proof that SCT is wrong does not challenge the original thesis. However, the Strong Church–Turing thesis is still widely accepted as an axiom that underlies theoretical computer science, and establishes a mathematical principle for computing analogous to those underlying physics and other sciences.

The equivalence of the Strong Church–Turing thesis to the original is a myth, clearly refuted by interactive models of computation. The widespread acceptance of this myth rests on the following beliefs:

1. All computable problems are mathematical problems expressible by functions from integers to integers, and therefore captured by Turing machines.
2. All computable problems can be described by algorithms (the primary form of all computation).
3. Algorithms are what computers do.

The first of these beliefs views computer science as a mathematical discipline. According to this world-view, mathematics strengthens the form of computing just as it has strengthened scientific models of physics and other disciplines. Though Turing was educated as a mathematician, he did not share the mathematical world-view [1]. However, mathematicians like Martin Davis, Von Neumann, Karp, Rabin, Scott, and Knuth accepted the mathematical ideas of Pythagoras, Descartes, Hilbert, and others that mathematics was an a priori rationalist principle that lay at the root of philosophy and science. They ignored Godel and Turing's proofs that mathematics was too week to be a universal problem solving principle in favor of the old a priori belief that mathematics was at the foundation of science in general and computer science in particular.

The second of these beliefs positions algorithms at the center of computer science; it ties the first and the third beliefs together, resulting in the Strong Church–Turing thesis. This central position of algorithms was a deliberate historical development of the 1960s, when the discipline of computer science was still in its formative stages. While there was an agreement on the strong role of algorithms, there was no agreement on their definition; two distinct and incompatible interpretations can be identified. The first interpretation, found in Knuth [5], defines algorithms as function-based transformations of inputs

to outputs; the second, found in less theoretical textbooks such as [8], defines them as abstract descriptions of the behavior of a program. Yuri Gurevich's chapter in this book [2] also reflects this second view of algorithms.

While the former interpretation of the notion of algorithm is consistent with the rationalist approach of the first belief, the latter interpretation is consistent with the empiricist approach of the third belief. The incompatibility of these interpretations pulls apart the three beliefs, bringing down the Strong Church–Turing thesis.

Hoare, Milner, and other Turing award winners realized in the 1970s that Turing machines do not model all problem solving, but believed it was not yet appropriate to challenge TMs as a complete model of computation. They separated interaction from computation, thereby avoiding the view that interaction was an expanded form of computation, raised by Wegner in 1997 [16].

The interactive view of computation is now widely accepted by many programmers, but is strongly disputed by adherents of the Turing machine model who regard the interaction model as an unnecessary and unproven paradigm shift. We believe it is now appropriate to accept the legitimacy of interactive models of computation, since new applications of agents, embedded systems, and the Internet expand the role of interaction as a fundamental part of computation.

8 Conclusion

Interaction provides an expanded model of computing that extends the class of computable problems from algorithms computable by Turing machines to interactive adaptive behavior of airline reservation systems or automatic cars. The paradigm shift from algorithms to interaction requires a change in modes of thought from a priori rationalism to empiricist testing that impacts scientific models of physics, mathematics, or computing, political models of human behavior, and religious models of belief. The substantive shift in modes of thought has led in the past to strong criticism by rationalist critics of empiricist models of Darwinian evolution or Galilean astronomy. Our chapter goes beyond the establishment of interaction as an extension of algorithms computable by Turing machines to the question of empiricist over rationalist modes of thought.

This chapter contributes to goals of this book by establishing interaction as an expanded form of computational problem solving, and to the exploration of principles that should underlie our acceptance of new modes of thought and behavior. Our section on persistent Turing machines (PTMs) examines the proof that sequential interaction is more expressive than Turing machine computation, while our section on the Church–Turing thesis shows that the Strong version of this thesis, with its assumption that Turing machines completely express computation, is both inaccurate and a denial of Turing's 1936 paper.

Our chapter has been influenced by Russell's *History of Western Philosophy* [10], whose articles on Descartes, Kant, and other philosophers support our philosophical arguments, and by Kuhn, whose book on scientific revolutions [6] supports the view that paradigm changes in scientific disciplines may require changes in modes of thought about the nature of truth.

References

1. E. Eberbach, D. Goldin, P. Wegner. Turing's Ideas and Models of Computation. In *Alan Turing: Life and Legacy of a Great Thinker*, ed. Christof Teuscher. Springer 2004.
2. Y. Gurevich. Interactive Algorithms 2005. In current book.
3. D. Goldin, S. Smolka, P. Attie, E. Sonderegger. Turing Machines, Transition Systems, and Interaction. *Information & Computation J.*, Nov. 2004.
4. D. Goldin, P. Wegner. The Church-Turing Thesis: Breaking the Myth. *LNCS 3526*, Springer, June 2005, pp. 152-168.
5. D. Knuth. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*. Addison-Wesley, 1968.
6. T. S. Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, 1962.
7. R. Penrose. *The Emperor's New Mind*, Oxford, 1989.
8. J. K. Rice, J. N. Rice. *Computer Science: Problems, Algorithms, Languages, Information and Computers.* Holt, Rinehart and Winston, 1969.
9. M. Prasse, P. Rittgen. Why Church's Thesis Still Holds - Some Notes on Peter Wegner's Tracts on Interaction and Computability, *Computer Journal* 41:6, 1998, pp. 357–362.
10. B. Russell. *History of Western Philosophy*. Simon and Schuster, 1945.
11. C. Teuscher, editor. *Alan Turing: Life and Legacy of a Great Thinker*. Springer 2004
12. A. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem, *Proc. London Math. Soc.*, 42:2, 1936, pp. 230-265; A correction, ibid, 43, 1937, 544–546.
13. A. Turing. Systems of logic based on ordinals, *Proc. London Math. Soc.*, 45:2, 1939, 161–228.
14. A. Turing. Computing Machinery and Intelligence, *Mind*, 1950.
15. J. van Leeuwen, J. Wiedermann. A Theory of Interactive Computation. In current book.
16. P. Wegner. Why Interaction is More Powerful Than Algorithms. *Comm. ACM*, May 1997.
17. P. Wegner. Interactive Foundations of Computing. *Theoretical Computer Science* 192, Feb. 1998.

http://www.springer.com/978-3-540-34666-1

Interactive Computation
The New Paradigm
Goldin, D.; Smolka, S.A.; Wegner, P. (Eds.)
2006, XV, 487 p. 84 illus., Hardcover
ISBN: 978-3-540-34666-1