The Blackwell Guide to the Philosophy of Computing and Information

Edited by

Luciano Floridi

Blackwell Publishing

350 Main Street, Malden, MA 02148-5020, USA
108 Cowley Road, Oxford OX4 1JF, UK
550 Swanston Street, Carlton, Victoria 3053, Australia

First published 2004 by Blackwell Publishing Ltd

Library of Congress Cataloging-in-Publication Data

The Blackwell guide to the philosophy of computing and information / edited by Luciano Floridi.
p. cm. — (Blackwell philosophy guides)
Includes bibliographical references and index.
ISBN 0-631-22918-3 (alk. paper) — ISBN 0-631-22919-1 (pbk. : alk. paper)
1. Computer science—Philosophy. 2. Information technology—Philosophy.
I. Floridi, Luciano, 1964– II. Series.

QA76.167.B53 2003
004′.01—dc21
2003045339

A catalogue record for this title is available from the British Library.

Set in 9/11.5pt Galliard
by Graphicraft Limited, Hong Kong
Printed and bound in the United Kingdom
by MPG Books Ltd, Bodmin, Cornwall

For further information on
Blackwell Publishing, visit our website:
http://www.blackwellpublishing.com

Contents

Chapter 1

Computation

B. Jack Copeland

The Birth of the Modern Computer

As everyone who can operate a personal computer knows, the way to make the machine perform some desired task is to open the appropriate program stored in the computer's memory. Life was not always so simple. The earliest large-scale electronic digital computers, the British Colossus (1943) and the American ENIAC (1945), did not store programs in memory (see Copeland 2001). To set up these computers for a fresh task, it was necessary to modify some of the machine's wiring, rerouting cables by hand and setting switches. The basic principle of the modern computer – the idea of controlling the machine's operations by means of a program of coded instructions stored in the computer's memory – was thought of by Alan Turing in 1935. His abstract "universal computing machine," soon known simply as the universal Turing machine (UTM), consists of a limitless memory, in which both data and instructions are stored, and a scanner that moves back and forth through the memory, symbol by symbol, reading what it finds and writing further symbols. By inserting different programs into the memory, the machine is made to carry out different computations.

Turing's idea of a universal stored-program computing machine was promulgated in the US by John von Neumann and in the UK by Max Newman, the two mathematicians who were by and large responsible for placing Turing's abstract universal machine into the hands of electronic engineers (Copeland 2001). By 1945, several groups in both countries had embarked on creating a universal Turing machine in hardware. The race to get the first electronic stored-program computer up and running was won by Manchester University where, in Newman's Computing Machine Laboratory, the "Manchester Baby" ran its first program on June 21, 1948. By 1951, electronic stored-program computers had begun to arrive in the marketplace. The first model to go on sale was the Ferranti Mark I, the production version of the Manchester computer (built by the Manchester firm Ferranti Ltd.). Nine of the Ferranti machines were sold, in Britain, Canada, Holland, and Italy, the first being installed at Manchester University in February 1951. In the US, the Computer Corporation sold its first UNIVAC later the same year. The LEO computer also made its debut in 1951; LEO was a commercial version of the prototype EDSAC machine, which at Cambridge University in 1949 had become the second stored-program electronic computer to function. In 1953 came the IBM 701, the company's first mass-produced stored-program electronic computer (strongly influenced by von Neumann's prototype IAS computer, which was working at

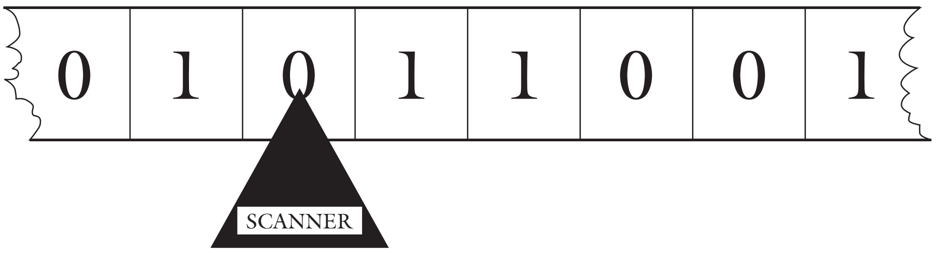

Figure 1.1: A Turing machine

Princeton University by the summer of 1951). A new era had begun.

Turing introduced his abstract Turing machines in a famous article entitled "On Computable Numbers, with an Application to the Entscheidungsproblem" (published in 1936). Turing referred to his abstract machines simply as "computing machines" – the American logician Alonzo Church dubbed them "Turing machines" (Church 1937: 43). "On Computable Numbers" pioneered the theory of computation and is regarded as the founding publication of the modern science of computing. In addition, Turing charted areas of mathematics lying beyond the reach of the UTM. He showed that not all precisely-stated mathematical problems can be solved by a Turing machine. One of them is the *Entscheidungsproblem* – "decision problem" – described below. This discovery wreaked havoc with received mathematical and philosophical opinion. Turing's work – together with contemporaneous work by Church (1936a, 1936b) – initiated the important branch of mathematical logic that investigates and codifies problems "too hard" to be solvable by Turing machine. In a single article, Turing ushered in both the modern computer and the mathematical study of the uncomputable.

What is a Turing Machine?

A Turing machine consists of a limitless memory and a scanner that moves back and forth through the memory, symbol by symbol, reading what it finds and writing further symbols. The memory consists of a tape divided into squares. Each square may be blank or may bear a single symbol, "0" or "1," for example, or some other symbol taken from a finite alphabet. The scanner is able to examine only one square of tape at a time (the "scanned square"). (See figure 1.1.) The tape is the machine's general-purpose storage medium, serving as the vehicle for input and output, and as a working memory for storing the results of intermediate steps of the computation. The tape may also contain a program of instructions. The input that is inscribed on the tape before the computation starts must consist of a finite number of symbols. However, the tape itself is of unbounded length – since Turing's aim was to show that there are tasks which these machines are unable to perform, even given unlimited working memory and unlimited time. (A Turing machine with a tape of fixed finite length is called a *finite state automaton*. The theory of finite state automata is not covered in this chapter. An introduction may be found in Sipser 1997.)

The Basic Operations of a Turing Machine

Each Turing machine has the same small repertoire of *basic* (or "atomic") operations. These are logically simple. The scanner contains mechanisms that enable it to *erase* the symbol on the scanned square, to *write* a symbol on the scanned square (first erasing any existing symbol), and to *shift position* one square to the left or right. Complexity of operation is achieved by chaining together large numbers of these simple

Table 1.1

State	*Scanned square*	*Operations*	*Next state*
a	blank	P[0], R	**b**
b	blank	R	**c**
c	blank	P[1], R	**d**
d	blank	R	**a**

basic actions. The scanner will *halt* if instructed to do so, i.e. will cease work, coming to rest on some particular square, for example the square containing the output (or if the output consists of a string of several digits, then on the square containing the left-most digit of the output, say).

In addition to the operations just mentioned, *erase*, *write*, *shift*, and *halt*, the scanner is able to *change state*. A device within the scanner is capable of adopting a number of different positions. This device may be conceptualized as consisting of a dial with a finite number of positions, labeled "a," "b," "c," etc. Each of these positions counts as a different state, and changing state amounts to shifting the dial's pointer from one labeled position to another. The device functions as a simple memory. As Turing said, by altering its state the "machine can effectively remember some of the symbols which it has 'seen' (scanned) previously" (1936: 231). For example, a dial with two positions can be used to keep a record of which binary digit, 0 or 1, is present on the square that the scanner has just vacated. If a square might also be blank, then a dial with three positions is required.

Commercially available computers are hard-wired to perform basic operations considerably more sophisticated than those of a Turing machine – add, multiply, decrement, store-at-address, branch, and so forth. The precise list of basic operations varies from manufacturer to manufacturer. It is a remarkable fact that none of these computers can out-compute the UTM. Despite the austere simplicity of Turing's machines, they are capable of computing anything that any computer on the market can compute. Indeed, because they are abstract machines, they are capable of computations that no "real" computer could perform.

Example of a Turing machine

The following simple example is from "On Computable Numbers" (Turing 1936: 233). The machine – call it **M** – starts work with a blank tape. The tape is endless. The problem is to set up the machine so that if the scanner is positioned over any square of the tape and the machine set in motion, it will print alternating binary digits on the tape, 0 1 0 1 0 1 . . . , working to the right from its starting place, leaving a blank square in between each digit. In order to do its work **M** makes use of four states labeled "**a**," "**b**," "**c**," and "**d**." **M** is in state **a** when it starts work. The operations that **M** is to perform can be set out by means of a table with four columns (see table 1.1). "R" abbreviates the instruction "shift right one square," "P[0]" abbreviates "print 0 on the scanned square," and likewise "P[1]." The top line of table 1.1 reads: if you are in state **a** and the square you are scanning is blank, then print 0 on the scanned square, shift right one square, and go into state **b**. A machine acting in accordance with this table of instructions – or program – toils endlessly on, printing the desired sequence of digits while leaving alternate squares blank.

Turing did not explain how it is to be brought about that the machine acts in accordance with the instructions. There was no need. Turing's machines are abstractions and it is not necessary to propose any specific mechanism for causing the machine to follow the instructions. However, for purposes of visualization, one might imagine the scanner to be accompanied by a bank of switches and plugs resembling an old-fashioned telephone switchboard. Arranging the plugs and setting the switches in a certain way causes the machine to act in accordance with the instructions in table 1.1. Other ways of setting up the "switchboard" cause the machine to act in accordance with other tables of instructions.

The universal Turing machine

The UTM has a single, fixed table of instructions, which we may imagine to have been set into the machine by way of the switchboard-like arrangement just mentioned. Operating in accordance with this table of instructions, the UTM is able to carry out *any* task for which a Turing-machine instruction table can be written. The trick is to place an instruction table for carrying out the desired task onto the tape of the universal machine, the first line of the table occupying the first so many squares of the tape, the second line the next so many squares, and so on. The UTM reads the instructions and carries them out on its tape. This ingenious idea is fundamental to computer science. The universal Turing machine is in concept the stored-program digital computer.

Turing's greatest contributions to the development of the modern computer were:

- The idea of controlling the function of the computing machine by storing a program of (symbolically or numerically encoded) instructions in the machine's memory.
- His proof that, by this means, a *single* machine of *fixed structure* is able to carry out every computation that can be carried out by any Turing machine whatsoever.

Human Computation

When Turing wrote "On Computable Numbers," a computer was not a machine at all, but a human being – a mathematical assistant who calculated by rote, in accordance with some "effective method" supplied by an overseer prior to the calculation. A paper-and-pencil method is said to be effective, in the mathematical sense, if it (a) demands no insight or ingenuity from the human carrying it out, and (b) produces the correct answer in a finite number of steps. (An example of an effective method well-known among philosophers is the truth table test for tautologousness.) Many thousands of human computers were employed in business, government, and research establishments, doing some of the sorts of calculating work that nowadays is performed by electronic computers. Like filing clerks, computers might have little detailed knowledge of the end to which their work was directed.

The term "computing machine" was used to refer to calculating machines that mechanized elements of the human computer's work. These were in effect homunculi, calculating more quickly than an unassisted human computer, but doing nothing that could not in principle be done by a human clerk working effectively. Early computing machines were somewhat like today's nonprogrammable hand-calculators: they were not automatic, and each step – each addition, division, and so on – was initiated manually by the human operator. For a complex calculation, several dozen human computers might be required, each equipped with a desk-top computing machine. By the 1940s, however, the scale of some calculations required by physicists and engineers had become so great that the work could not easily be done in a reasonable time by even a roomful of human computers with desk-top computing machines. The need to develop high-speed, large-scale, automatic computing machinery was pressing.

In the late 1940s and early 1950s, with the advent of electronic computing machines, the phrase "computing machine" gave way gradually to "computer." During the brief period in which the old and new meanings of "computer" co-existed, the prefix "electronic" or "digital" would usually be used in order to distinguish machine from human. As Turing stated, the new electronic machines were "intended to carry out any definite rule of thumb process which could have been done by a human operator working in a disciplined but unintelligent manner" (Turing 1950: 1). Main-frames, laptops, pocket calculators, palm-pilots – all carry out work that a human rote-worker could do, if he or she worked long enough, and had a plentiful enough supply of paper and pencils.

The Turing machine is an idealization of the human computer (Turing 1936: 231). Wittgenstein put this point in a striking way:

> Turing's "Machines." These machines are *humans* who calculate. (Wittgenstein 1980: §1096)

It was not, of course, some deficiency of imagination that led Turing to model his logical computing machines on what can be achieved by a human being working effectively. The purpose for which he introduced them demanded it. The Turing machine played a key role in his demonstration that there are mathematical tasks which *cannot* be carried out by means of an effective method.

The Church–Turing Thesis

The concept of an effective method is an informal one. Attempts such as the above to explain what counts as an effective method are not rigorous, since the requirement that the method demand neither insight nor ingenuity is left unexplicated. One of Turing's leading achievements – and this was a large first step in the development of the mathematical theory of computation – was to propose a rigorously defined expression with which the informal expression "by means of an effective method" might be replaced. The rigorously defined expression, of course, is "by means of a Turing machine." The importance of Turing's proposal is this: if the proposal is correct, then talk about the existence and non-existence of effective methods can be replaced throughout mathematics and logic by talk about the existence or non-existence of Turing machine programs. For instance, one can establish that there is no effective method at all for doing such-and-such a thing by proving that no Turing machine can do the thing in question.

Turing's proposal is encapsulated in the *Church–Turing thesis*, also known simply as *Turing's thesis*:

> The UTM is able to perform any calculation that any human computer can carry out.

An equivalent way of stating the thesis is:

> Any effective – or *mechanical* – method can be carried out by the UTM.

("Mechanical" is a term of art in mathematics and logic. It does not carry its everyday meaning, being in its technical sense simply a synonym for "effective.") Notice that the converse of the thesis – any problem-solving method that can be carried out by the UTM is effective – is obviously true, since a human being can, in principle, work through any Turing-machine program, obeying the instructions ("in principle" because we have to assume that the human does not go crazy with boredom, or die of old age, or use up every sheet of paper in the universe).

Church independently proposed a different way of replacing talk about effective methods with formally precise language (Church 1936a). Turing remarked that his own way of proceeding was "possibly more convincing" (1937: 153); Church acknowledged the point, saying that Turing's concept of computation by Turing machine "has the advantage of making the identification with effectiveness . . . evident immediately" (Church 1937: 43).

The name "Church–Turing thesis," now standard, seems to have been introduced by Kleene, with a flourish of bias in favor of his mentor Church (Kleene 1967: 232):

> Turing's and Church's theses are equivalent. We shall usually refer to them both as *Church's thesis*, or in connection with that one of its . . . versions which deals with "Turing machines" as *the Church–Turing thesis*.

Soon ample evidence amassed for the Church–Turing thesis. (A survey is given in chs. 12 and 13 of Kleene 1952.) Before long it was (as Turing put it) "agreed amongst logicians" that his proposal gives the "correct accurate rendering" of talk about effective methods (Turing 1948: 7). (Nevertheless, there have been occasional dissenting voices over the years; for example Kalmár 1959 and Péter 1959.)

Beyond the Universal Turing Machine

Computable and uncomputable numbers

Turing calls any number that can be written out by a Turing machine a *computable* number. That is, a number is computable, in Turing's sense, if and only if there is a Turing machine that calculates each digit of the number's decimal representation, in sequence. π, for example, is a computable number. A suitably programmed Turing machine will spend all eternity writing out the decimal representation of π digit by digit, 3.14159 . . .

Straight off, one might expect it to be the case that *every* number that *has* a decimal representation (that is to say, every real number) is computable. For what could prevent there being, for any particular number, a Turing machine that "churns out" that number's decimal representation digit by digit? However, Turing proved that not every real number is computable. In fact, computable numbers are relatively scarce among the real numbers. There are only *countably* many computable numbers, because there are only countably many different Turing-machine programs (instruction tables). (A collection of things is countable if and only if either the collection is finite or its members can be put into a one-to-one correspondence with the integers, 1, 2, 3, ...) As Georg Cantor proved in 1874, there are *uncountably* many real numbers – in other words, there are more real numbers than integers. There are literally not enough Turing-machine programs to go around in order for every real number to be computable.

The printing problem and the halting problem

Turing described a number of mathematical problems that cannot be solved by Turing machine. One is the *printing problem.* Some programs print "0" at some stage in their computations; all the remaining programs never print "0." The printing problem is the problem of deciding, given any arbitrarily selected program, into which of these two categories it falls. Turing showed that this problem cannot be solved by the UTM.

The *halting problem* (Davis 1958) is another example of a problem that cannot be solved by the UTM (although not one explicitly considered by Turing). This is the problem of determining, given any arbitrary Turing machine, whether or not the machine will eventually halt when started on a blank tape. The machine shown in table 1.1 is rather obviously one of those that never halts – but in other cases it is definitely not obvious from a machine's table whether or not it halts. And, of course, simply watching the machine run (or a simulation of the machine) is of no help at all, for what can be concluded if after a week or a year the machine has not halted? If the machine does eventually halt, a watching human – or Turing machine – will sooner or later find this out; but in the case of a machine that has not yet halted, there is no effective method for deciding whether or not it is going to halt.

The halting function

A *function* is a mapping from "arguments" (or inputs) to "values" (or outputs). For example, addition (+) is a function that maps pairs of numbers to single numbers: the value of the function + for the pair of arguments 5, 7 is the number 12. The squaring function maps single numbers to single numbers: e.g. the value of n^2 for the argument 3 is 9.

A function is said to be *computable by Turing machine* if some Turing machine will take in arguments of the function (or pairs of arguments, etc.) and, after carrying out some finite number of basic operations, produce the corresponding value – and, moreover, will do this *no matter which* argument of the function is presented. For example, addition over the integers is computable by Turing machine, since a Turing machine can be set up so that whenever two integers are inscribed on its tape (in binary notation, say), the machine will output their sum.

The *halting function* is as follows. Assume the Turing machines to be ordered in some way, so that we may speak of the first machine in the ordering, the second, and so on. (There are various standard ways of accomplishing this ordering, e.g. in terms of the number of symbols in each machine's instruction table.) The arguments of the halting function are simply 1, 2, 3, ... (Like the squaring function, the halting function takes single arguments.) The value of the halting function for any argument *n* is 1 if the nth Turing machine in the ordering eventually halts when started on a blank tape, and is 0 if the nth machine runs on forever (as would, for example, a Turing machine programmed to produce in succession the digits of the decimal representation of π).

The theorem that the UTM cannot solve the halting problem is often expressed in terms of the halting function.

Halting theorem: The halting function is not computable by Turing machine.

The Entscheidungsproblem

The *Entscheidungsproblem*, or decision problem, was Turing's principal quarry in "On Computable Numbers." The decision problem was brought to the fore of mathematics by the German mathematician David Hilbert (who in a lecture given in Paris in 1900 set the agenda for much of twentieth-century mathematics). Hilbert and his followers held that mathematicians should seek to express mathematics in the form of a complete, consistent, decidable formal system – a system expressing "the entire thought-content of mathematics in a uniform way" (Hilbert 1927: 475). The project of formulating mathematics in this way became known as the "Hilbert program."

A consistent system is one that contains no contradictions; a complete system one in which every true mathematical statement is provable. "Decidable" means that there is an effective method for telling, of each mathematical statement, whether or not the statement is provable in the system. A complete, consistent, decidable system would banish ignorance from mathematics. Given any mathematical statement, one would be able to tell whether the statement is true or false by deciding whether or not it is provable in the system. Hilbert famously declared in his Paris lecture: "in mathematics there is no *ignorabimus*" (there is no *we shall not know*) (Hilbert 1902: 445).

It is important that the system expressing the "whole thought content of mathematics" be consistent. An inconsistent system – a system containing contradictions – is worthless, since *any* statement whatsoever, true or false, can be derived from a contradiction by simple logical steps. So in an inconsistent system, absurdities such as $0 = 1$ and $6 \neq 6$ are provable. An inconsistent system would indeed contain all true mathematical statements – would be complete, in other words – but would in addition also contain all false mathematical statements.

If ignorance is to be banished absolutely, the system must be decidable. An undecidable system might on occasion leave us in ignorance. Only if the mathematical system were decidable could we be confident of always being able to tell whether or not any given statement is provable. Unfortunately for the Hilbert program, however, it became clear that most interesting mathematical systems are, if consistent, incomplete and undecidable.

In 1931 Gödel showed that Hilbert's ideal is impossible to satisfy, even in the case of simple arithmetic. He proved that the system called Peano arithmetic is, if consistent, incomplete. This is known as Gödel's *first incompleteness theorem*. (Gödel later generalized this result, pointing out that "due to A. M. Turing's work, a precise and unquestionably adequate definition of the general concept of formal system can now be given," with the consequence that incompleteness can "be proved rigorously for *every* consistent formal system containing a certain amount of finitary number theory" (Gödel 1965: 71).) Gödel had shown that no matter how hard mathematicians might try to construct the all-encompassing formal system envisaged by Hilbert, the product of their labors would, if consistent, inevitably be incomplete. As Hermann Weyl – one of Hilbert's greatest pupils – observed, this was nothing less than "a catastrophe" for the Hilbert program (Weyl 1944: 644).

Gödel's theorem does not mention decidability. This aspect was addressed by Turing and by Church. Each showed, working independently, that no consistent formal system of arithmetic is decidable. They showed this by proving that not even the weaker, purely logical system presupposed by any formal system of arithmetic and called the *first-order predicate calculus* is decidable. Turing's way of proving that the first-order predicate calculus is undecidable involved the printing problem. He showed that if a Turing machine could tell, of any given statement, whether or not the statement is provable in the first-order predicate calculus, then a Turing machine could tell, of any given Turing machine, whether or not it ever prints "0." Since, as he had already established, no Turing machine can do the latter, it follows that no Turing machine can do the former. The final step of the argument is to apply Turing's thesis: if no Turing machine can perform the task in question, then there is no effective method for performing it. The Hilbertian dream lay in total ruin.

Poor news though Turing's and Church's result was for the Hilbert school, it was welcome news in other quarters, for a reason that Hilbert's illustrious pupil von Neumann had given in 1927 (von Neumann 1927: 12):

> If undecidability were to fail then mathematics, in today's sense, would cease to exist; its place would be taken by a completely mechanical rule, with the aid of which any man would be able to decide, of any given statement, whether the statement can be proven or not.

In a similar vein, the Cambridge mathematician G. H. Hardy said in a lecture in 1928 (Hardy 1929: 16):

> if there were . . . a mechanical set of rules for the solution of all mathematical problems . . . our activities as mathematicians would come to an end.

The next section is based on Copeland 1996.

Misunderstandings of the Church–Turing Thesis: The Limits of Machines

A myth has arisen concerning Turing's work, namely that he gave a treatment of the limits of mechanism, and established a fundamental result to the effect that the UTM can simulate the behavior of *any* machine. The myth has passed into the philosophy of mind, theoretical psychology, cognitive science, Artificial Intelligence, and Artificial Life, generally to pernicious effect. For example, the *Oxford Companion to the Mind* states: "Turing showed that his very simple machine . . . can specify the steps required for the solution of any problem that can be solved by instructions, explicitly stated rules, or procedures" (Gregory 1987: 784). Dennett maintains that "Turing had proven – and this is probably his greatest contribution – that his Universal Turing machine can compute any function that any computer, with any architecture, can compute" (1991: 215); also that every "task for which there is a clear recipe composed of simple steps can be performed by a very simple computer, a universal Turing machine, the universal recipe-follower" (1978: xviii). Paul and Patricia Churchland assert that Turing's "results entail something remarkable, namely that a standard digital computer, given only the right program, a large enough memory and sufficient time, can compute *any* rule-governed input–output function. That is, it can display any systematic pattern of responses to the environment whatsoever" (1990: 26). Even Turing's biographer, Hodges, has endorsed the myth:

> Alan had . . . discovered something almost . . . miraculous, the idea of a universal machine that could take over the work of *any* machine. (Hodges 1992: 109)

Turing did not show that his machines can solve any problem that can be solved "by instructions, explicitly stated rules, or procedures," and nor did he prove that the UTM "can compute any function that any computer, with any architecture, can compute" or perform any "task for which there is a clear recipe composed of simple steps." As previously explained, what he proved is that the UTM can carry out any task that any *Turing machine* can carry out. Each of the claims just quoted says considerably more than this.

If what the Churchlands assert were true, then the view that psychology must be capable of being expressed in standard computational terms would be secure (as would a number of other controversial claims). But Turing had no result entailing that "a standard digital computer . . . can compute *any* rule-governed input–output function." What he did have was a result entailing the exact opposite. The theorem that no Turing machine can decide the predicate calculus entails that there are rule-governed input–output functions that no Turing machine is able to compute – for example, the function whose output is 1 whenever the input is a statement that is provable in the predicate calculus, and is 0 for all other inputs. There are certainly possible patterns of responses to the environment, perfectly systematic patterns, that no Turing machine can display. One is the pattern of responses just described. The halting function is a mathematical characterization of another such pattern.

Distant cousins of the Church–Turing thesis

As has already been emphasized, the Church–Turing thesis concerns the extent of effective methods. Putting this another way (and ignoring contingencies such as boredom, death, or insufficiency of paper), the thesis concerns what a *human being* can achieve when working by rote with paper and pencil. The thesis carries no implication concerning the extent of what *machines* are capable of achieving (even digital machines acting in accordance with "explicitly stated rules"). For among a machine's repertoire of basic operations, there may be those that no human working by rote with paper and pencil can perform.

Essentially, then, the Church–Turing thesis says that no human computer, or machine that mimics a human computer, can out-compute the UTM. However, a variety of other propositions, very different from this, are from time to time called the Church–Turing thesis (or Church's thesis), sometimes but not always with accompanying hedges such as "strong form" and "physical version." Some examples from the recent literature are given below. This loosening of established terminology is unfortunate, and can easily lead to misunderstandings. In what follows I use the expression "Church–Turing thesis properly so called" for the proposition that Turing and Church themselves endorsed.

> [C]onnectionist models . . . may possibly even challenge the strong construal of Church's Thesis as the claim that the class of well-defined computations is exhausted by those of Turing machines. (Smolensky 1988: 3)

> Church–Turing thesis: If there is a well defined procedure for manipulating symbols, then a Turing machine can be designed to do the procedure. (Henry 1993: 149)

> [I]t is difficult to see how any language that could actually be run on a physical computer could do more than Fortran can do. The idea that there is no such language is called Church's thesis. (Geroch & Hartle 1986: 539)

> The first aspect that we examine of Church's Thesis . . . [w]e can formulate, more precisely: The behaviour of any discrete physical system evolving according to local mechanical laws is recursive. (Odifreddi 1989: 107)

> I can now state the physical version of the Church–Turing principle: "Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means." This formulation is both better defined and more physical than Turing's own way of expressing it. (Deutsch 1985: 99)

> That there exists a most general formulation of machine and that it leads to a unique set of input–output functions has come to be called *Church's thesis*. (Newell 1980: 150)

The maximality thesis

It is important to distinguish between the Church–Turing thesis properly so called and what I call the "maximality thesis" (Copeland 2000). (Among the few writers to distinguish explicitly between Turing's thesis and stronger propositions along the lines of the maximality thesis are Gandy 1980 and Sieg 1994.)

A machine *m* is said to be able to *generate* a certain function if *m* can be set up so that if *m* is presented with any of the function's arguments, *m* will carry out some finite number of atomic processing steps at the end of which *m* produces the corresponding value of the function (*mutatis mutandis* in the case of functions that, like addition, demand more than one argument).

> **Maximality Thesis**: All functions that can be generated by machines (working on finite input in accordance with a finite program of instructions) are computable by Turing machine.

The maximality thesis ("thesis M") admits of two interpretations, according to whether the phrase "can be generated by machine" is taken in the this-worldly sense of "can be generated by a machine that conforms to the physical laws (if not to the resource constraints) of the actual world," or in a sense that abstracts from whether or not the envisaged machine could exist in the actual world. Under the latter interpretation, thesis M is false. It is straightforward to describe abstract machines that generate functions that cannot be generated by the UTM (see e.g. Abramson 1971, Copeland 2000, Copeland & Proudfoot 2000, Stewart 1991). Such machines are termed "hypercomputers" in Copeland and Proudfoot (1999a).

It is an open empirical question whether or not the this-worldly version of thesis M is true. Speculation that there may be physical processes – and so, potentially, machine-operations – whose behavior conforms to functions not computable by Turing machine stretches back over at least five decades. (Copeland & Sylvan 1999 is a survey; see also Copeland & Proudfoot 1999b.)

A source of potential misunderstanding about the limits of machines lies in the difference between the technical and everyday meanings of the word "mechanical." As previously remarked, in technical contexts "mechanical" and "effective" are often used interchangeably. (Gandy 1988 outlines the history of this usage of the word "mechanical.") For example:

> Turing proposed that a certain class of abstract machines could perform any "mechanical" computing procedure. (Mendelson 1964: 229)

Understood correctly, this remark attributes to Turing not a thesis concerning the limits of what can be achieved by machine but the Church–Turing thesis properly so called.

The technical usage of "mechanical" tends to obscure the possibility that there may be machines, or biological organs, that generate (or compute, in a broad sense) functions that cannot be computed by Turing machine. For the question "Can a machine execute a procedure that is not mechanical?" may appear self-answering, yet this is precisely what is asked if thesis M is questioned.

In the technical literature, the word "computable" is often tied by definition to effectiveness: a function is said to be computable if and only if there is an effective method for determining its values. The Church–Turing thesis then becomes:

> Every computable function can be computed by Turing machine.

Corollaries such as the following are sometimes stated:

> [C]ertain functions are uncomputable in an absolute sense: uncomputable even by [Turing machine], and, therefore, uncomputable by any past, present, or future real machine. (Boolos & Jeffrey 1980: 55)

When understood in the sense in which it is intended, this remark is perfectly true. However, to a casual reader of the technical literature, such statements may appear to say more than they in fact do.

Of course, the decision to tie the term "computable" and its cognates to the concept of effectiveness does not settle the truth-value of thesis M. Those who abide by this terminological decision will not describe a machine that falsifies thesis M as *computing* the function that it generates.

Putnam is one of the few writers on the philosophy of mind to question the proposition that Turing machines provide a maximally general formulation of the notion of machine:

> [M]aterialists are committed to the view that a human being is – at least metaphorically – a machine. It is understandable that the notion of a Turing machine might be seen as just a

> way of making this materialist idea precise. Understandable, but hardly well thought out. The problem is the following: a "machine" in the sense of a physical system obeying the laws of Newtonian physics need not be a Turing machine. (Putnam 1992: 4)

The Church–Turing fallacy

To commit what I call the *Church–Turing fallacy* (Copeland 2000, 1998) is to believe that the Church–Turing thesis, or some formal or semi-formal result established by Turing or Church, secures the following proposition:

> If the mind–brain is a machine, then the Turing-machine computable functions provide sufficient mathematical resources for a full account of human cognition.

Perhaps some who commit this fallacy are misled purely by the terminological practice already mentioned, whereby a thesis concerning which there is little real doubt, the Church–Turing thesis properly so called, and a nexus of different theses, some of unknown truth-value, are all referred to as Church's thesis or the Church–Turing thesis.

The Church–Turing fallacy has led to some remarkable claims in the foundations of psychology. For example, one frequently encounters the view that psychology *must* be capable of being expressed ultimately in terms of the Turing machine (e.g. Fodor 1981: 130; Boden 1988: 259). To anyone in the grip of the Church–Turing fallacy, conceptual space will seem to contain no room for mechanical models of the mind–brain that are not equivalent to a Turing machine. Yet it is certainly possible that psychology will find the need to employ models of human cognition that transcend Turing machines (see Chapter 10, COMPUTATIONALISM, CONNECTIONISM, AND THE PHILOSOPHY OF MIND).

The simulation fallacy

A closely related error, unfortunately also common in modern writing on computation and the brain, is to hold that Turing's results somehow entail that the brain, and indeed any biological or physical system whatever, can be *simulated* by a Turing machine. For example, the entry on Turing in *A Companion to the Philosophy of Mind* contains the following claims: "we can depend on there being a Turing machine that captures the functional relations of the brain," for so long as "these relations between input and output are functionally well-behaved enough to be describable by . . . mathematical relationships . . . we know that some specific version of a Turing machine will be able to mimic them" (Guttenplan 1994: 595). Even Dreyfus, in the course of *criticizing* the view that "man is a Turing machine," succumbs to the belief that it is a "fundamental truth that every form of 'information processing' (even those which *in practice* can only be carried out on an 'analogue computer') must *in principle* be simulable on a [Turing machine]" (1992: 195).

Searle writes in a similar fashion:

> If the question ["Is consciousness computable?"] asks "Is there some level of description at which conscious processes and their correlated brain processes can be simulated [by a Turing machine]?" the answer is trivially yes. Anything that can be described as a precise series of steps can be simulated [by a Turing machine]. (Searle 1997: 87)

> Can the operations of the brain be simulated on a digital computer? . . . The answer seems to me . . . demonstrably "Yes" . . . That is, naturally interpreted, the question means: Is there some description of the brain such that under that description you could do a computational simulation of the operations of the brain. But given Church's thesis that anything that can be given a precise enough characterization as a set of steps can be simulated on a digital computer, it follows trivially that the question has an affirmative answer. (Searle 1992: 200)

Church's thesis properly so called does *not* say that anything that can be described as a precise series of of steps can be simulated by Turing machine.

Similarly, Johnson-Laird and the Churchlands argue:

> If you assume that [consciousness] is scientifically explicable . . . [and] [g]ranted that the [Church–Turing] thesis is correct, then the final dichotomy rests on Craik's functionalism. If you believe [functionalism] to be false . . . then presumably you hold that consciousness could be modelled in a computer program in the same way that, say, the weather can be modelled . . . If you accept functionalism, however, then you should believe that consciousness is a computational process. (Johnson-Laird 1987: 252)

> Church's Thesis says that whatever is computable is Turing computable. Assuming, with some safety, that what the mind-brain does is computable, then it can in principle be simulated by a computer. (Churchland & Churchland 1983: 6)

As previously mentioned, the Churchlands believe, incorrectly, that Turing's "results entail . . . that a standard digital computer, given only the right program, a large enough memory and sufficient time, can . . . display any systematic pattern of responses to the environment whatsoever" (1990: 26). This no doubt explains why they think they can assume "with some safety" that what the mind–brain does is computable, for on their understanding of matters, this is to assume only that the mind–brain is characterized by a "rule-governed" (1990: 26) input–output function.

The Church–Turing thesis properly so called does not entail that the brain (or the mind, or consciousness) can be simulated by a Turing machine, not even in conjunction with the belief that the brain (or mind, etc.) is scientifically explicable, or exhibits a systematic pattern of responses to the environment, or is "rule-governed" (etc.). Each of the authors quoted seems to be assuming the truth of a close relative of thesis M, which I call "thesis S" (Copeland 2000).

> **Thesis S**: Any process that can be given a mathematical description (or a "precise enough characterization as a set of steps," or that is scientifically describable or scientifically explicable) can be simulated by a Turing machine.

As with thesis M, thesis S is trivially false if it is taken to concern all conceivable processes, and its truth-value is unknown if it is taken to concern only processes that conform to the physics of the real world. For all we presently know, a completed neuroscience may present the mind–brain as a machine that – when abstracted out from sources of inessential boundedness, such as mortality – generates functions that no Turing machine can generate.

The equivalence fallacy

Paramount among the evidence for the Church–Turing thesis properly so called is the fact that all attempts to give an exact analysis of the intuitive notion of an effective method have turned out to be *equivalent*, in the sense that each analysis has been proved to pick out the same class of functions, namely those that are computable by Turing machine. (For example, there have been analyses in terms of lambda-definability, recursivenes, register machines, Post's canonical and normal systems, combinatory definability, Markov algorithms, and Gödel's notion of reckonability.) Because of the diversity of these various analyses, their equivalence is generally considered very strong evidence for the Church–Turing thesis (although for a skeptical point of view see Kreisel 1965: 144).

However, the equivalence of these diverse analyses is sometimes taken to be evidence also for stronger theses like M and S. This is nothing more than a confusion – the *equivalence fallacy* (Copeland 2000). The analyses under discussion are of the notion of an effective method, not of the notion of a machine-generable function; the equivalence of the analyses bears only on the issue of the extent of the former notion and indicates nothing concerning the extent of the latter.

Artificial intelligence and the equivalence fallacy

Newell, discussing the possibility of artificial intelligence, argues that (what he calls) a "physical symbol system" can be organized to exhibit general intelligence. A "physical symbol system" is a universal Turing machine, or any equivalent system, situated in the physical – as opposed to the conceptual – world. (The tape of the machine is accordingly finite; Newell specifies that the storage capacity of the tape [or equivalent] be unlimited in the practical sense of finite yet not small enough to "force concern.")

> A [physical symbol] system always contains the potential for being any other system if so instructed. Thus, a [physical symbol] system can become a generally intelligent system. (Newell 1980: 170)

Is the premise of this pro-AI argument true? A physical symbol system, being a *universal* Turing machine situated in the real world, can, if suitably instructed, simulate (or, metaphorically, become) any other physical symbol system (*modulo* some fine print concerning storage capacity). If this is what the premise means, then it is true. However, if taken literally, the premise is false, since as previously remarked, systems can be specified which no Turing machine – and so no physical symbol system – can simulate. However, if the premise is interpreted in the former manner, so that it is true, the conclusion fails to follow from the premise. Only to one who believes, as Newell does, that "the notion of machine or determinate physical mechanism" is "formalized" by the notion of a Turing machine (ibid.) will the argument appear deductively valid.

Newell's defense of his view that the universal Turing machine exhausts the possibilities of mechanism involves an example of the equivalence fallacy:

> [An] important chapter in the theory of computing . . . has shown that all attempts to . . . formulate . . . general notions of mechanism . . . lead to classes of machines that are equivalent in that they encompass in toto exactly the same set of input–output functions. In effect, there is a single large frog pond of functions no matter what species of frogs (types of machines) is used. . . . A large zoo of different formulations of maximal classes of machines is known by now – Turing machines, recursive functions, Post canonical systems, Markov algorithms . . . (Newell 1980: 150)

Newell's *a priori* argument for the claim that a physical symbol system can become generally intelligent founders in confusion.

Conclusion

Since there are problems that cannot be solved by Turing machine, there are – given the Church–Turing thesis – limits to what can be accomplished by any form of machine that works in accordance with effective methods. However, not all *possible* machines share those limits. It is an open empirical question whether there are actual deterministic physical processes that, in the long run, elude simulation by Turing machine; and, if so, whether any such processes could usefully be harnessed in some form of calculating machine. It is, furthermore, an open empirical question whether any such processes are involved in the working of the human brain.

References

Abramson, F. G. 1971. "Effective computation over the real numbers." *Twelfth Annual Symposium on Switching and Automata Theory.* Northridge, CA: Institute of Electrical and Electronics Engineers.

Boden, M. A. 1988. *Computer Models of Mind.* Cambridge: Cambridge University Press.

Boolos, G. S. and Jeffrey, R. C. 1980. *Computability and Logic*, 2nd ed. Cambridge: Cambridge University Press.

Church, A. 1936a. "An unsolvable problem of elementary number theory." *American Journal of Mathematics* 58: 345–63.

——. 1936b. "A note on the Entscheidungsproblem." *Journal of Symbolic Logic* 1: 40–1.

——. 1937. Review of Turing 1936. *Journal of Symbolic Logic* 2: 42–3.

Churchland, P. M. and Churchland, P. S. 1983. "Stalking the wild epistemic engine." *Nous* 17: 5–18.

—— and ——. 1990. "Could a machine think?" *Scientific American* 262 (Jan.): 26–31.

Copeland, B. J. 1996. "The Church–Turing Thesis." In E. Zalta, ed., *The Stanford Encyclopaedia of Philosophy*, <http://plato.stanford.edu>.

——. 1998. "Turing's O-machines, Penrose, Searle, and the Brain." *Analysis* 58: 128–38.

——. 2000. "Narrow versus wide mechanism, including a re-examination of Turing's views on the mind–machine issue." *Journal of Philosophy* 97: 5–32. Repr. in M. Scheutz, ed., *Computationalism: New Directions.* Cambridge, MA: MIT Press, 2002.

——. 2001. "Colossus and the dawning of the computer age." In M. Smith and R. Erskine, eds., *Action This Day.* London: Bantam.

——. and Proudfoot, D. 1999a. "Alan Turing's forgotten ideas in computer science." *Scientific American* 280 (April): 76–81.

—— and ——. 1999b. "The legacy of Alan Turing." *Mind* 108: 187–95.

—— and ——. 2000. "What Turing did after he invented the universal Turing machine." *Journal of Logic, Language, and Information* 9: 491–509.

—— and Sylvan, R. 1999. "Beyond the universal Turing machine." *Australasian Journal of Philosophy* 77: 46–66.

Davis, M. 1958. *Computability and Unsolvability.* New York: McGraw-Hill.

Dennett, D. C. 1978. *Brainstorms: Philosophical Essays on Mind and Psychology.* Brighton: Harvester.

——. 1991. *Consciousness Explained.* Boston: Little, Brown.

Deutsch, D. 1985. "Quantum theory, the Church–Turing principle and the universal quantum computer." *Proceedings of the Royal Society*, Series A, 400: 97–117.

Dreyfus, H. L. 1992. *What Computers Still Can't Do: A Critique of Artificial Reason.* Cambridge, MA: MIT Press.

Fodor, J. A. 1981. "The mind–body problem." *Scientific American* 244 (Jan.): 124–32.

Gandy, R. 1980. "Church's thesis and principles for mechanisms." In J. Barwise, H. Keisler, and K. Kunen, eds., *The Kleene Symposium.* Amsterdam: North-Holland.

——. 1988. "The confluence of ideas in 1936." In R. Herken, ed., *The Universal Turing Machine: A Half-century Survey.* Oxford: Oxford University Press.

Geroch, R. and Hartle, J. B. 1986. "Computability and physical theories." *Foundations of Physics* 16: 533–50.

Gödel, K. 1965. "Postscriptum." In M. Davis, ed., *The Undecidable.* New York: Raven, pp. 71–3.

Gregory, R. L. 1987. *The Oxford Companion to the Mind.* Oxford: Oxford University Press.

Guttenplan, S. 1994. *A Companion to the Philosophy of Mind.* Oxford: Blackwell.

Hardy, G. H. 1929. "Mathematical proof." *Mind* 38: 1–25.

Henry, G. C. 1993. *The Mechanism and Freedom of Logic.* Lanham, MD: University Press of America.

Hilbert, D. 1902. "Mathematical problems: lecture delivered before the International Congress of Mathematicians at Paris in 1900." *Bulletin of the American Mathematical Society* 8: 437–79.

——. 1927. "Die Grundlagen der Mathematik" [The Foundations of Mathematics]. English trans. in J. van Heijenoort, ed., *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931.* Cambridge, MA: Harvard University Press, 1967.

Hodges, A. 1992. *Alan Turing: The Enigma.* London: Vintage.

Johnson-Laird, P. 1987. "How could consciousness arise from the computations of the brain?" In C. Blakemore and S. Greenfield, eds., *Mindwaves.* Oxford: Blackwell.

Kalmár, L. 1959. "An argument against the plausibility of Church's thesis." In A. Heyting, ed., *Constructivity in Mathematics.* Amsterdam: North-Holland.

Kleene, S. C. 1952. *Introduction to Metamathematics.* Amsterdam: North-Holland.

——. 1967. *Mathematical Logic.* New York: Wiley.

Kreisel, G. 1965. "Mathematical logic." In T. L. Saaty, ed., *Lectures on Modern Mathematics*, vol. 3. New York: Wiley.

Langton, C. R. 1989. "Artificial life." In Langton, ed., *Artificial Life.* Redwood City: Addison-Wesley.

Mendelson, E. 1964. *Introduction to Mathematical Logic.* New York: Van Nostrand.

Newell, A. 1980. "Physical symbol systems." *Cognitive Science* 4: 135–83.

Odifreddi, P. 1989. *Classical Recursion Theory.* Amsterdam: North-Holland.

Péter, R. 1959. "Rekursivität und Konstruktivität." In A. Heyting, ed., *Constructivity in Mathematics.* Amsterdam: North-Holland.

Putnam, H. 1992. *Renewing Philosophy.* Cambridge, MA: Harvard University Press.

Searle, J. 1992. *The Rediscovery of the Mind.* Cambridge, MA: MIT Press.

——. 1997. *The Mystery of Consciousness.* New York: New York Review of Books.

Sieg, W. 1994. "Mechanical procedures and mathematical experience." In A. George, ed., *Mathematics and Mind*. Oxford: Oxford University Press.

Sipser, M. 1997. *Introduction to the Theory of Computation*. Boston: PWS Publishing.

Smolensky, P. 1988. "On the proper treatment of connectionism." *Behavioral and Brain Sciences* 11: 1–23.

Stewart, I. 1991. "Deciding the undecidable." *Nature* 352: 664–5.

Turing, A. M. 1936. "On computable numbers, with an application to the Entscheidungsproblem." *Proceedings of the London Mathematical Society*, series 2, 42 (1936–7): 230–65.

——. 1937. "Computability and λ-definability." *Journal of Symbolic Logic* 2: 156–63.

——. 1948. "Intelligent machinery." National Physical Laboratory Report. In B. Meltzer and D. Michie, eds., *Machine Intelligence 5.* Edinburgh: Edinburgh University Press, 1969. [A digital facsimile is available in The Turing Archive for the History of Computing, <http://www.AlanTuring.net/intelligent_machinery>.]

——. 1950. "Programmers' handbook for Manchester electronic computer." University of Manchester Computing Laboratory. [A digital facsimile is available in The Turing Archive for the History of Computing, <http://www.AlanTuring. net/programmers_handbook>.]

von Neumann, J. 1927. "Zur Hilbertschen Beweistheorie" [On Hilbert's proof theory], *Mathematische Zeitschrift* 26: 1–46.

Weyl, H. 1944. "David Hilbert and his Mathematical Work," *Bulletin of the American Mathematical Society* 50: 612–54.

Wittgenstein, L. 1980. *Remarks on the Philosophy of Psychology*, vol. 1. Oxford: Blackwell.

Chapter 25

Philosophy of Information Technology

Carl Mitcham

Philosophy of information technology may be seen as a special case of the philosophy of technology. Philosophical reflection on technology aims in general to comprehend the nature and meaning of the making and using, especially of things made and used. Such reflection nevertheless exhibits a tension between two major traditions: one arising within engineering, another in the humanities (Mitcham 1994). For the former or expansionist view, technology is deeply and comprehensively human, and thus properly extended into all areas of life; according to the latter or limitationist perspective, technology is a restricted and properly circumscribed dimension of the human. This distinction and corresponding tensions may also be seen at play in the philosophy of information technology (IT), between those who would critically celebrate and extend IT and those who would cautiously subordinate and delimit it. Diverse metaphysical, epistemological, and ethical arguments are marshaled to defend one position over the other, as well as to build bridges between these two philosophical poles.

Philosophies of *x* commonly begin with attempts to define *x*. Philosophy of science, for instance, logically opens with the demarcation problem, by considering various proposals for distinguishing science from other forms of knowledge or human activity. The philosophy of information technology, like the philosophy of computer science, is properly initiated by the effort to define that on which it seeks to reflect. Once preliminary definitions are negotiated, philosophies of *x*, often against a historico-philosophical background, recapitulate in differentially weighted forms the main branches of philosophy *tout court* – metaphysics, epistemology, and ethics – with particular emphases reflecting both the unique philosophical challenges of *x* and the context of presentation. In the present case, for example, although ethical and political issues play a prominent role in the philosophy of information technology, they are treated lightly here because of the more extensive coverage provided by Chapters 5 and 6 (Computer Ethics and Computer-mediated Communication and Human–Computer Interaction). Here the stress is on theoretical issues concerning especially metaphysical assessments of information technology.

What Is Information Technology?

Information technology – or such closely related terms as "information systems" and "media technology" – is commonly described as that technology constituted by the merging of data-processing and telecommunications (with diverse input devices, processing programs, communications systems, storage formats, and output displays). It arose from earlier forms of electronic communications technology (telegraph, telephone, phonograph, radio, motion pictures, television) by way of computers and cybernetics (see Chapter 14), an earlier term that still casts its shadow over IT, as in such coinages as "cyberspace" and other cognates. It may nevertheless be useful to begin by attempting to rethink what is perhaps too facile in such a description.

The terms "information" and "technology" are both subject to narrow and broad, not to say engineering and humanities, definitions. Developed by Claude Shannon (Shannon & Weaver 1949), the technical concept of information is defined as the probability of a signal being transmitted from device A to device B, which can be mathematically quantified (see Chapters 4 and 13, treating INFORMATION and THE PHYSICS OF INFORMATION, respectively). The theory of information opened up by this distinct conceptual analysis has become the basis for constructing and analyzing digital computational devices and a whole range of information (also called communication) technologies, from telephones to televisions and the internet.

In contrast to information (and information technologies) in the technical sense is the concept of information in a broader or semantic sense. Semantic information is not a two-term relation – that is, a signal being transmitted from A to B – but a three-term relation: a signal transmitted from one device to another, which is then understood as saying something to a person C. Although information technologies in the technical sense readily become information technologies in the semantic sense, there is no precise relation between technical and semantic information. Independent of its probability as a signal, some particular transmission may possess any number of different semantic meanings. A signal in the form of two short clicks or light flashes (Morse Code for the letter "i"), could be a self-referential pronoun, part of the word "in," a notation in Latin numerals of the number one – or any number of other possibilities. Absent the context, a signal is not a message. Kenneth Sayre (1976) and Fred I. Dretske (1981) are nevertheless two important attempts to develop semantic theories of information grounded in the technical concept of information (see Chapter 17, KNOWLEDGE).

"Technology" too is a term with narrow and broad definitions. In the narrow or engineering sense, technology is constituted by the systematic study and practice of the making and using of artifacts (cf. the curricula of technological universities), and to some extent by the physical artifacts themselves (from hammers to cars and computers). Indeed, a distinction is often drawn between premodern *techne* or technics and modern technology. For thousands of years human making and using proceeded by intuitive, trial-and-error methods, remained mostly small-scale and dwarfed by natural phenomena. With the rise of modern methods for making and using, these activities became systematically pursued (often on the basis of scientific theories) and their products began to rival natural phenomena in scale and scope. In a broader humanities parlance, technology covers both intuitive, small-scale and scientific, large-scale making and using in all its modes – as knowledge, as artifact, as activity, and even as volition.

Given these narrow and broad definitions for each element in the compound term, one may postulate a two by two matrix and imagine four different information technology exemplars (figure 25.1). In what follows, a significant sample from among these possible information technologies will be analyzed in order to illustrate diverse facets of a potentially comprehensive philosophy of information technology.

Information Technology in Historico-philosophical Perspective

Philosophy is not coeval with human thought, but emerges from and against prephilosophical reflection that it nevertheless continues or mirrors. Prior to the rise of philosophy, mythological and poetic narratives often expressed the ambivalence of the human experience of tool making

	Premodern technology	*Scientific technology*
Technical sense of information	Alphabetic writing	Electronic and source code signal transmission
Semantic sense of information	Books and related texts	Works of high representational electronic communications media (movies, TV programs, hypertexts, etc.)

Figure 25.1: Information technology exemplars

and using. Stories of the conflict between Cain (builder of cities) and Abel (pastoral shepherd), of Prometheus (who stole fire for humans from the gods), of Hephaestus (the deformed god of the forge), and of Icarus (the inventor who went too far) all attest to the problematic character of human engagement with what has come to be called technology. The story of the Tower of Babel (Genesis 11) even suggests the destructive linguistic repercussions of an excessive pursuit of technological prowess.

By contrast, when the prophet Ezekiel learns in the desert to infuse dry bones (alphabetic consonants) with the breath of the spirit (unwritten vowels), it is as if God were speaking directly through him (Ezekiel 37). Indeed, God himself creates through speech or *logos* (Genesis 1), and writes the law both in stone and in the hearts of a people. Thus, information technologies in their earliest forms – speech and writing – manifest at least two fundamental experiences of the human condition: sin or hubris and transcendence, the demonic and the divine.

Greek philosophical reflection on *techne* likewise noted the two-fold tendency of human skill in the making and using of artifacts to be pursued in isolation from the good and to participate in the divine. This is as true of information *technai*, such as oratory and writing, as it is of the mechanical and military arts. In Plato's *Gorgias*, for instance, Socrates challenges the sophist to reintegrate the techniques of rhetoric with the pursuit of truth, to eschew the tricks of gaining power divorced from knowledge of the good. In the *Phaedrus*, Socrates tells the story of how King Thamus rejected the Egyptian god Theuth's invention of writing on the grounds that it would replace real with merely virtual memory (*Phaedrus* 274d ff.). Socrates himself comments on the silence of written words, and Plato famously remarked on the limitations of writing even in his own works (*Letter* VII, 341b–e and 344c–d). The *Politicus* (300c ff.), however, concludes with a modest defense of written laws, and the *Ion* presents the poet as one inspired by the gods.

Aristotle, in an analysis that echoes Plato's assessment in *The Republic* of artifice and poetry as thrice-removed from being itself, notes the inability of *techne* to effect a substantial unity of form and matter. "If a bed were to sprout," says Aristotle, "not a bed would come up but a tree" (*Physics* II, i, 193a12–16). In a parallel analysis of the relation between experiences, spoken words, and writing at the beginning of *On Interpretation*, Aristotle places the written word at two removes from experience and three removes from the things experienced, thus implying a dilution of contact with reality as one moves from the information technology of speech to that of writing. Spoken words refer to experience; written words to spoken words.

In contrast to Aristotle's characterization of words in strictly human terms, Christianity reaffirms the divine character of the transcendent word incarnate (John 1) and of the transmission of the gospel through that preaching which represents the word (Romans 10:17). Indeed, according to Augustine, Christian preaching unites truth and language with an efficacy that the Platonists could not imagine (*De vera religione* i, 1 ff.). This is an argument that has been revived in Catherine Pickstock's theological interpretation of that information technology known as liturgy (Pickstock 1998). At the same time, the meaning of the words of revelation in Scripture is not always obvious, thus requiring the development of principles of interpretation (see Augustine's *De doctrina christiana*). Faith in the Scriptures as the word of God solves, as it were, the technical question concerning the extent to which the signal has been accurately transmitted from A to B (God to humans), but not the semantic question of what this signal means (to whom it speaks and about what). The meaning of revelation requires a science of interpretation or hermeneutics if its information (from the Latin *informare*, to give form) is truly to convert those who receive it.

This dedication to the development of techniques of interpretation led to a unique medieval flowering of logical, rhetorical, and hermeneutic prowess. Reflecting the effulgence of poetic exegesis of sacred texts, Thomas Aquinas defends the metaphorical "hiding of truth in figures" as fitting to the word of God, and argues the power of Scripture to signify by way of multiple references: historical or literal, allegorical, tropological or moral, and anagogical or eschatological (*Summa theologiae* I, q.1, art.9–10). What is equally remarkable is that – no doubt stimulated by the literal and spiritual interpretations of revelation as granting the world a certain autonomy and calling upon human beings to exercise positive mastery over it – the flowering of semantic studies was paralleled by an equally unprecedented blossoming of physical technologies. Examples include the waterwheel and windmill, the moldboard plow, the horse collar, the lateen sail, and the mechanical clock.

The modern world opens, paradoxically, by pitting the second form of technological progress (physical inventions) against the first (poetic creativity). Metaphorical words are to be rejected in the pursuit of real things and ever more powerful technologies (see especially the arguments of Francis Bacon and René Descartes). The historical result was to turn exegesis into criticism and semantic analysis into a drive for conceptual clarity, in a reform of the techniques of communication that became most manifest in the new rhetoric of modern natural science – as well as in the invention of a whole new information technology known as moveable type.

The invention of the printing press and the consequent democratization of reading can be associated with a manifold of social transformations: religious, political, economic, and cultural. The philosophical influences of such changes have been legion. To cite but one example, as the world was increasingly filled with texts, and texts themselves were severed from stable lifeworlds of interpreters, philosophy became increasingly linguistic philosophy, in two forms. In continental Europe, hermeneutics was redefined by Friedrich Schleiermacher as the interpretation of all (not just sacred) texts, by Wilhelm Dilthey as the foundation of the *Geistwissenschaften* or humanistic sciences, and by Martin Heidegger as the essence of *Dasein* or human being. In this same milieu, Ferdinand de Saussure invented the science of linguistics, focusing neither on efficient signal transmission nor on multiple levels of external reference but on language as a system of words that mutually define one another through their internal relations. In the Anglo-American world, especially under the influence of Ludwig Wittgenstein, philosophy became linguistic philosophy, which takes the meaning of words to be constituted by their uses, thus calling attention to multiple contexts of use, what Wittgenstein called ways of life. Indeed, in some forms the resultant philosophy of language turns into a kind of behaviorism or is able to make common cause with pragmatism.

In another instance, theories were posited about the relation between changes in information technologies and cultural orders. The contrast between orality and literacy has been elaborated by a series of scholars – from Albert Lord and Milman Parry to Marshall McLuhan, Walter Ong, and Ivan Illich – who have posited complementary theories about relations between information technology transformations and cultures. With McLuhan, for example, there is a turn not just from technical signal to semantic message, but an attempt to look at the whole new electronic signal transmitting and receiving technology (never mind any specific semantic content) as itself a message. In his own condensed formulation: the medium (or particular form of information technology) is the message (McLuhan 1964).

Stimulated especially by McLuhan, reviews of the historical influences between philosophy and IT begin to mesh into a philosophy of history that privileges IT experience the way G. W. F. Hegel privileged politics and ideas. Here Paul Levinson's "natural history of information technology" (1997) is a worthy illustration.

Information Technology and Metaphysics

Although the historico-philosophical background points to an emergence, in conjunction with information technology, of new cultural constellations in human affairs, pointing alone is insufficient to constitute philosophy. Popular attempts to think the new IT lifeworld have emphasized economics and politics, in which issues are decided about e-banking and e-commerce on the basis of market forces and political power. The ethics of information technology, as an initiation into philosophical reflection – that is, into thought in which issues are assessed on the basis of argument and insight rather than money and votes – has highlighted issues of privacy, equity, and accountability. Yet given that the fundamental question for ethics concerns how to act in accord with what really is, there are reasons to inquire into the kind of reality disclosed by IT – that is, to raise metaphysical (beyond the physical or empirical) and ontological (from *ontos*, the Greek word for "being") questions.

What are the fundamental structures of the IT phenomenon? What is real and what is appearance with regard to IT? Richard Coyne (1995), for instance, argues that it is illusory to view IT as simply a novel instrument available for the effective realization of traditional projects for conserving and manipulating data. Albert Borgmann (1999) insightfully distinguishes between information about reality (science), information for reality (engineering design), and information as reality (the high-definition representations and creations emerging from IT) – and further the increasing prominence, glamor, and malleability of information as reality is having the effect of diminishing human engagement with more fundamental realities. With regard to the kinds of metaphysical issues raised by Coyne, Borgmann, and others, it is useful to distinguish again expansionist and limitationist approaches to the nature and meaning for information technology.

The expansionist approach has its roots in technical thinking about IT, first in terms of physical entities. At least since Norbert Wiener (1948) effectively posited that, along with matter and energy, information is a fundamental constituent of reality, questions have been raised about the metaphysical status of information. Building on Wiener's own analysis, distinctions may be drawn between three fundamentally different kinds of technology: those which transform matter (hammers and assembly lines), those which produce and transform energy (power plants and motors), and those which transform information (communication systems and computers).

A related phenomenology of human engagement would observe how the being of IT differs from tools and machines. Unlike tools (which do not function without human energy input and guidance) or machines (which derive energy from nonhuman sources but still require human guidance), information technologies are in distinctive ways independent of the human with regard to energy and immediate guidance; they are self-regulating (cybernetic). In this sense, steam engines with mechanical governors on them or thermostatically controlled heating systems are examples of information machines. Insofar as the operation of more electronically advanced IT is subject to human guidance, guidance ceases to be direct or mechanical and is mediated by humanly constructed programs (electronically coded plans). What is the ontological status of programs? What are their relations to intentions? Indeed, in IT, operation and use appear to have become distinguishable. IT is a new species of artifact, a hybrid that is part machine running on its own and part utility structure like a road waiting to be driven on – hence the term "new media" (as both means and environment). The static availability of such structures is contingent on their semi-autonomous dynamic functioning.

Second, in terms of the cognitive capabilities of IT, transempirical questions arise about the extent to which computers (as pervasive elements in IT) imitate human cognitive processes. Do computers think? What kind of intelligence is artificial intelligence (AI)? Are the different kinds of AI – algorithmic, heuristic, connectionist, embodied, etc. – different forms of intelligence? Such ontological questions now blend into others, concerning the extent to which high-tech artifacts are different from living organisms. Biotechnology has breached Aristotle's distinction between natural tree and artificial bed, growth and construction, the born and the made. Soon computer programs may also be able not just to mimic patterns of growth on the screen (artificial life, see Chapter 15), but autonomous, artificial agents that are able to reproduce themselves. At the nano-scale, robotic design will hardly be distinct from genetic engineering. Will any differences in being remain?

From the technical perspective, information is ubiquitous in both the organic and the artificial worlds. The wall between the two is vanishing, although, insofar as the technical concept of information becomes a category of explanation in biology, it has also been argued to have distinct ideological roots (see Kay 2000 on this point). The cyborg (cybernetic organism) is a living machine, not a goddess (Haraway 1991). Within such a reality, the ethical imperative becomes experimenting with ourselves, what Coyne (1995) calls a pragmatic interaction with advancing IT. This is an attitude widely present among leading IT designers such as Mark Weiser at the famous Xerox Palo Alto Research Center (PARC), the ethos of which is commonly celebrated in *Wired* magazine. It has also have been given philosophical articulation by media philosopher Wolfgang Schirmacher. For Schirmacher (1994), IT is a kind of artificial nature, a post-technology in which we are free (and obligated, if we would act in harmony with the new way of being in the world) to live without predeterminations, playfully and aesthetically.

The limitationist approach originates in a different, more skeptical stance. Issues are no doubt oversimplified by characterizing one approach as pro-IT and another as con-IT – although such a contrast captures some measure of real difference (but see Gordon Graham, 1999, for a down-to-earth philosophical utilization of this contrast using the terms technophiles and neoluddites). Perhaps a better contrast would be that of Hegel versus Socrates: the comprehensive critical affirmation as opposed to the argumentative gadfly. From the Hegelian perspective there is something both adolescent and irresponsible about an ongoing Socratic negativity that refuses to take responsibility for world creation. Indeed, Socratic negativity easily becomes a philosophically clichéd substitute for true thinking. From the Socratic perspective, however, the expansionist approach comes on the scene as a court philosophy, especially insofar as it flatters the king and counsels expanding an already popular and widely affirmed domain of influence. In a state already dominated by information technology, the Socratic tradition thus finds expression in repeatedly questioning the nature and meaning of IT – a questioning that must ultimately go metaphysical.

At a first level, however, the questioning of IT will be, as already suggested, ethical. For instance, does IT not threaten privacy? Even more profoundly, does the IT mediation of human action in complex software programs, which are created by multiple technicians and are not even in principle able to be fully tested (Zimmerli 1986), not challenge the very notion of moral accountability? At a second level are political questions: Is the internet structured so as to promote social justice through equity of access? Is it compatible with democracy? Furthermore, IT exists on the back of a substantial industrial base, whose environmental sustainability is at least debatable. Insofar as IT depends on an unsustainable base, might not its own justice and goodness be compromised? At still a third level are psychological questions, blending into epistemological ones. Does the exponential growth of information availability not challenge the human ability to make sense of it? Information overload or information anxiety (see Wurman 2001) is one of the most widely cited paradoxes of IT life. Finally, at a fourth level are psychological-anthropological questions about the social implications of the new "mode of information" (Poster 1990), what it means to live a "virtual life" (Brook & Boal 1995) and "life on the screen" (Turkle 1995).

The third and fourth dimensions of limitationist, Socratic questioning – that is, the epistemological and anthropological levels – hint at the metaphysical. Information technology may hide reality from us in a much more fundamental way than simply by means of information overload. It may deform our being at deeper levels than the psychological. To develop this possibility it is useful to refer at some length to Martin Heidegger, the most influential exponent of this position.

According to Heidegger's highly influential argument in "The Question Concerning Technology" (1977 [1954]), technology is constituted not so much by machines or even instrumental means in general as by its disclosure of reality, its unhiding, its truth. Premodern technology in the form of *poiesis* functioned as a bringing or leading forth that worked with nature, and as such revealed Being as alive with its own bringing forth, the way a seed blossoms into a flower or an acorn grows into an oak tree. Modern technology, by contrast, is not so much a bringing forth as a challenging-forth that reveals the world as *Bestand* or manipulatable resource.

In reading Heidegger it is crucial to recognize that he felt it necessary to couch his insights in a special vocabulary ("bringing forth," "challenging forth," "*Bestand*"), because of the way ordinary concepts are sedimented with assumptions that themselves help conceal the dimensions of reality to which he invites attention. In Heidegger's own words:

> The revealing that rules throughout modern technology has the character of a setting-upon, in the sense of a challenging-forth. That challenging happens in that the energy concealed in nature is unlocked, what is unlocked is transformed, what is transformed is stored up, what is stored up is, in turn, distributed, and what is distributed is switched about ever anew. Unlocking, transforming, storing, distributing, and switching about are ways of revealing. (Heidegger 1977 [1954]: 297–8)

To this distinctive way of revealing Heidegger also gives a special name: *Gestell* or enframing.

Although Heidegger seems to be thinking here of electric power generation, the same description would in many ways be applicable to information technology. There is a challenging that happens when digitally concealed information is unlocked (from, say, a computer disk), transformed (by some software program), stored up (on a hard drive), distributed (by internet), and switched about (forwarded, reprocessed, data mined, etc.). Indeed, in another text Heidegger makes the reference to IT explicit, although under the name of cybernetics. "Cybernetics," he writes, "transforms language into an exchange of news. The arts become regulated-regulating instruments of information" (Heidegger 1977 [1966]: 376). Modern information technology thus does to language what modern non-information technology does to the material world: turns it into *Bestand*, that is, a resource for human manipulation.

What is wrong with this? The basic answer is that modern technology, including modern information technology, conceals as well as it reveals. Insofar as we persist in emphasizing the revealing and ignore the concealing, concealing will actually dominate. We will not be fully aware of what is going on. To develop this point requires a brief elaboration of Heidegger's theory of hermeneutics. In his version of hermeneutics, which argues interpretation (more than rationality) as the defining characteristic of the human, Heidegger makes two basic claims.

The first is that no revealing (the acquisition of information in the semantic sense) is ever simple; it always involves the process of interpretation. Interpretation itself proceeds in texts, in perception, in thinking, and in life by means of a dialectic between part and whole, what is called the hermeneutic circle. The part is only revealed in terms of the whole, and the whole in terms of the parts. As a result, Heideggerian hermeneutics postulates a pregivenness in all revealing or, as he also likes to say, unconcealing. Our minds and our lives open not as with a *tabula rasa*, but with an immanent reality (both part and whole) waiting to be brought forth into the light of appearances. Understanding proceeds by means of a process of moving from part to whole and *vice versa*, repeatedly to make the implicit explicit, to reveal the concealed, analogous especially to the ways that premodern technology also worked to till the fields and to fashion handcrafted artifacts. The upshot is that not only is all information subject to interpretation, but that all information technology is part of a larger lifeworld and cannot be understood apart from such an implicit whole. To think otherwise is a metaphysical mistake.

The second claim is that any unconcealing is at one and the very same time a concealing. This second claim has even more profound implications for information technology, which through its expanding realms makes information more and more omnipresent. Information technology appears to reveal with a vengeance. According to Heidegger, however, this is ultimately an illusion – and dangerous to what it means to be human. The problem is not just one of sensory or information overload, but of information as a concealing of Being itself, the fundamental nature of reality, of the distinctly human relation to such reality.

For Heidegger the rise of modern technology, and its culmination in cybernetics or information technology, is the culmination of a historico-philosophical trajectory of thinking that began with the Greeks. With Plato and Aristotle, Being was first revealed, however tentatively and minimally, as a presence that could be re-presented in thought or rationalized. Over the course of its 2,500-year history, philosophy has successively spun off the various scientific disciplines as specialized ways to re-present the world: in mathematics, in logic, in astronomy, in physics, in chemistry, in biology, in cosmology, and now in the interdisciplinary fields of molecular biology, cognitive science, and more. This continuing development is the end of philosophy in two senses: its perfection and its termination. The very success of scientific revealing grew out of a specialization of thinking as philosophy that entailed leaving behind or concealing thinking in a more fundamental sense, something that Heidegger refers to as *Lichtung*, translated variously as "lighting" or "opening." "Perhaps there is a thinking," Heidegger writes, "which is more sober-minded than the incessant frenzy of rationalization and the intoxicating quality of cybernetics" (Heidegger 1977 [1966]: 391).

In another text, Heidegger describes this "new task of thinking" at "the end of philosophy" by means of a comparison between what he calls calculative and meditative thinking. "Calculative thinking never stops, never collects itself. Calculative thinking is not meditative thinking, not thinking which contemplates the meaning which reigns in everything that is" (Heidegger 1966 [1955]L 46). Meditative thinking, premodern and even preclassical Greek philosophical thinking, which was once in touch with the root of human existence, and out of which by means of a narrowing and intensified calculative thinking has emerged, has been replaced by calculative thinking in the form of "all that with which modern techniques of communication stimulate, assail, and drive human beings" (Heidegger 1966 [1955]: 48). Technology, especially information technology, conceals this meditative thinking, which Heidegger terms *Gelassenheit*, releasement or detachment. "Releasement toward things and openness to the mystery . . . promise us a new ground and foundation upon which we can stand and endure the world of technology without being imperiled by it" (Heidegger 1966 [1955]: 55). The fundamental threat in information technology is thus a threat to the human being's "essential nature" and the "issue of keeping meditative thinking alive" (ibid.: 56).

Current Research and Open Issues

What is most remarkable is the fact that Heidegger's radical critique of technology in general and information technology in particular has been subject to significant practical appropriations by IT users and designers, thus building bridges between the engineering and humanities, the expansionist and limitationist, traditions in the philosophy of information technology. Raphael Capurro (1986), for instance, brings Heidegger to bear on the field of library and information science. Hubert Dreyfus (2001) examines the Internet from a philosophical perspective indebted to Heidegger. With slightly more expansion, one may also reference two other leading examples: Terry Winograd and Fernando Flores, and Richard Coyne. At the same time serious challenges have been raised by Mark Poster to the adequacy of a Heideggerian approach to IT.

In the mid-1980s, computer scientists Winograd and Flores argued at length that Heideggerian analyses could disclose the reasons behind the failures of information technologies to function as well in the office as computer scientists predicted. In Winograd and Flores (1987) they argue that Heideggerian insights can thus be a stimulus for redesigning computer systems.

A decade later architectural theorist Coyne (1995) goes even further, arguing that not just Heidegger but the post-Heideggerian thought of Jacques Derrida provides a philosophical account of what is going on among leading-edge information technology designers. Building on Heidegger's notion that all revealing involves a simultaneous concealing, Derrida proposes to deconstruct specific concepts, methods, and disciplinary formations precisely to bring to light their hidden aspects, that on which they depend without knowing or acknowledging it. For Coyne this opens the way for and justifies the turn from a commitment to rational method in information technology design to the renewed reliance on metaphor.

Heidegger and Derrida thus revalidate the creative significance of metaphor – of thinking of a computer operating system as "windows," of a screen "desktop" with "icons," even of the mind as a computer. It is precisely a play with such "irrational" connections that facilitates advances in information technology design. With Aquinas, Coyne seeks to defend the metaphorical "hiding of truth in figures" as functional not just in theology but also in technology. Whether either Aquinas or Heidegger would counsel such appropriation of their philosophies of information technology is, of course, seriously in doubt.

As if to reinforce such doubt about such creative appropriations, Poster argues at length that Heidegger "captures the revealing of modern technology only, not postmodern technology." Indeed, "some information technologies, in their complex assemblages, partake not only of [*Gestell*] but also of forms of revealing that do not conceal but solicit participants to a relation to Being as freedom" (Poster 2001: 32–3). For Poster a more adequate approach to the philosophical understanding of IT is through Felix Guattari's image of the rhizome and a phenomenology of the enunciative properties of specific technologies. A potentially comprehensive philosophy of IT thus remains, not unlike all philosophy, suspended in and energized by its fundamental alternatives.

References

Borgmann, A. 1999. *Holding On To Reality: The Nature of Information at the Turn of the Millennium*. Chicago: University of Chicago Press. [Distinguishes natural information (about reality), cultural information (for constructing reality), and technological information (information becoming a reality in its own right). Seeks to establish guidelines for assessing and limiting information as reality.]

Brook, J. and Boal, I. A. 1995. *Resisting the Virtual Life: The Culture and Politics of Information*. San Francisco: City Lights. [Twenty-one critical essays on IT inequities, impacts on the body, the degrading of the workplace, and cultural deformations.]

Capurro, R. 1986. *Hermeneutik der Fachinformation*. Freiburg: Alber. [For a short English paper that reviews the thesis of this book, see R. Capurro, "Hermeneutics and the phenomenon of information," *Research in Philosophy and Technology* 19: 79–85.]

Coyne, R. 1995. *Designing Information Technology in the Postmodern Age: From Method to Metaphor*. Cambridge, MA: MIT Press. [Advances in information technology are examined from the diverse philosophical perspectives of analytic philosophy, pragmatism, phenomenology, critical theory, and hermeneutics, in order to reveal their different implications for the design and development of new electronic communications media.]

Dretske, F. 1983. *Knowledge and the Flow of Information*. Cambridge, MA: MIT Press. [The most well-developed theory of perception and empirical knowledge based on mathematical information theory.]

Dreyfus, H. L. 2001. *On the Internet*. New York: Routledge. [A phenomenologically influenced but interdisciplinary critique.]

Graham, G. 1999. *The Internet: A Philosophical Inquiry*. New York: Routledge. [Questions and criticizes both neoluddite and technophile claims about the dangers and implications of the internet.]

Haraway, D. 1991. *Simians, Cyborgs, and Women: The Reinvention of Nature*. New York: Routledge. [See especially the "Manifesto for Cyborgs" included in this book.]

Heidegger, M. 1966 [1955]. *Discourse on Thinking*, tr. J. M. Anderson and E. H. Freund. New York: Harper and Row. [This includes translation of Heidegger's essay, "Gelassenheit."]

——. 1977 [1954]. "The question concerning technology," tr. W. Lovitt. In M. Heidegger, *Basic Writings*. New York: Harper and Row, pp. 287–317. [Heidegger's most important critique of technology.]

——. 1977 [1966]. "The end of philosophy and the task of thinking," tr. J. Stambaugh. In M. Heidegger, *Basic Writings*. New York: Harper and Row, pp. 373–92. [A brief statement of Heidegger's philosophy of the history of philosophy.]

Kay, L. E. 2000. *Who Wrote the Book of Life: A History of the Genetic Code*. Stanford: Stanford University Press. [A critical assessment of information as a metaphor in biology.]

Levinson, P. 1997. *The Soft Edge: A Natural History and Future of the Information Revolution*. New York: Routledge. [A new information medium (such as computers) does not so much replace an old medium (such as the telephone) as complement it.]

McLuhan, M. 1964. *Understanding Media: The Extensions of Man*. New York: McGraw-Hill. [It is not the information content of a medium (such as speech or television) that is most influential on a culture, but the character or structure of the medium itself. Electronic media are structurally distinct from, say, books. "The medium is the message."]

Mitcham, C. 1994. *Thinking through Technology: The Path between Engineering and Philosophy*. Chicago: University of Chicago Press. [A general introduction to the philosophy of technology that distinguishes two major traditions: engineering and humanities philosophy of technology. The former argues for the expansion, the latter for the delimitation of technology as object, knowledge, activity, and volition.]

Pickstock, C. 1998. *After Writing: On the Liturgical Consummation of Philosophy*. Oxford, UK: Blackwell. [A critique of Derrida and defense of information as subordinate to the context created by linguistic and bodily performance in a historical tradition.]

Poster, Mark. 1990. *The Mode of Information: Poststructuralism and Social Context*. Chicago: University of Chicago Press. [Argues that four new modes of information – TV ads, data bases, electronic writing, and computer science – create a world in which humans are socially constituted differently than in pre-electronic IT history.]

Poster, Mark. 2001. *What's the Matter with the Internet*. Minneapolis, MI: University of Minnesota Press. [A critique of applying Heidegger's analysis of technology to information technology, with special reference to postmodern thinkers such as Felix Guattari.]

Sayre, K. M. 1976. *Cybernetics and the Philosophy of Mind*. London: Routledge & Kegan Paul. [Argues a naturalist theory of mind based on mathematical information theory.]

Schirmacher, W. 1994. "Media and postmodern technology." In G. Bender and T. Druckrey, eds., *Culture on the Brink: Ideologies of Technologies*. Seattle: Bay Press. [The other contributions to this book are useful as well.]

Shannon, C. and Weaver, W. 1949. *The Mathematical Theory of Communication*. Urbana: University of Illinois Press. [Contains two classic papers: Shannon's, from the *Bell System Technical Journal* (1948); and Weaver's, from *Scientific American* (1949).]

Turkle, Sherry. 1995. *Life on the Screen: Identity in the Age of the Internet*. New York: Simon & Schuster. [A psychologist's analysis of emerging forms of self-definition unique to the internet experience.]

Wiener, N. 1948. *Cybernetics: Or, Control and Communication in the Animal and the Machine*. Cambridge, MA: MIT Press. [The classic statement of the engineering theory of cybernetics. In other works Wiener also examined the social and ethical implications of his theories.]

Winograd, T. and Flores, F. 1987. *Understanding Computers and Cognition: A New Foundation for Design*. Reading, MA: Addison-Wesley. [A Heideggerian analysis by two computer scientists.]

Wurman, R. S. 2001. *Information Anxiety 2*. Indianapolis, IN: Que. [This is the second edition of a widely cited critique of information overload by a well-known architect and student of the work of Louis Kahn.]

Zimmerli, W. 1986. "Who is to blame for data pollution?" In C. Mitcham and A. Huning, eds., *Philosophy and Technology II: Information Technology and Computers in Theory and Practice*. Boston: Reidel. [One of 20 original papers from a conference, introduced by an overview of "Information technology and computers as themes in the philosophy of technology," and followed by an annotated bibliography on philosophical studies of information technology and computers.]