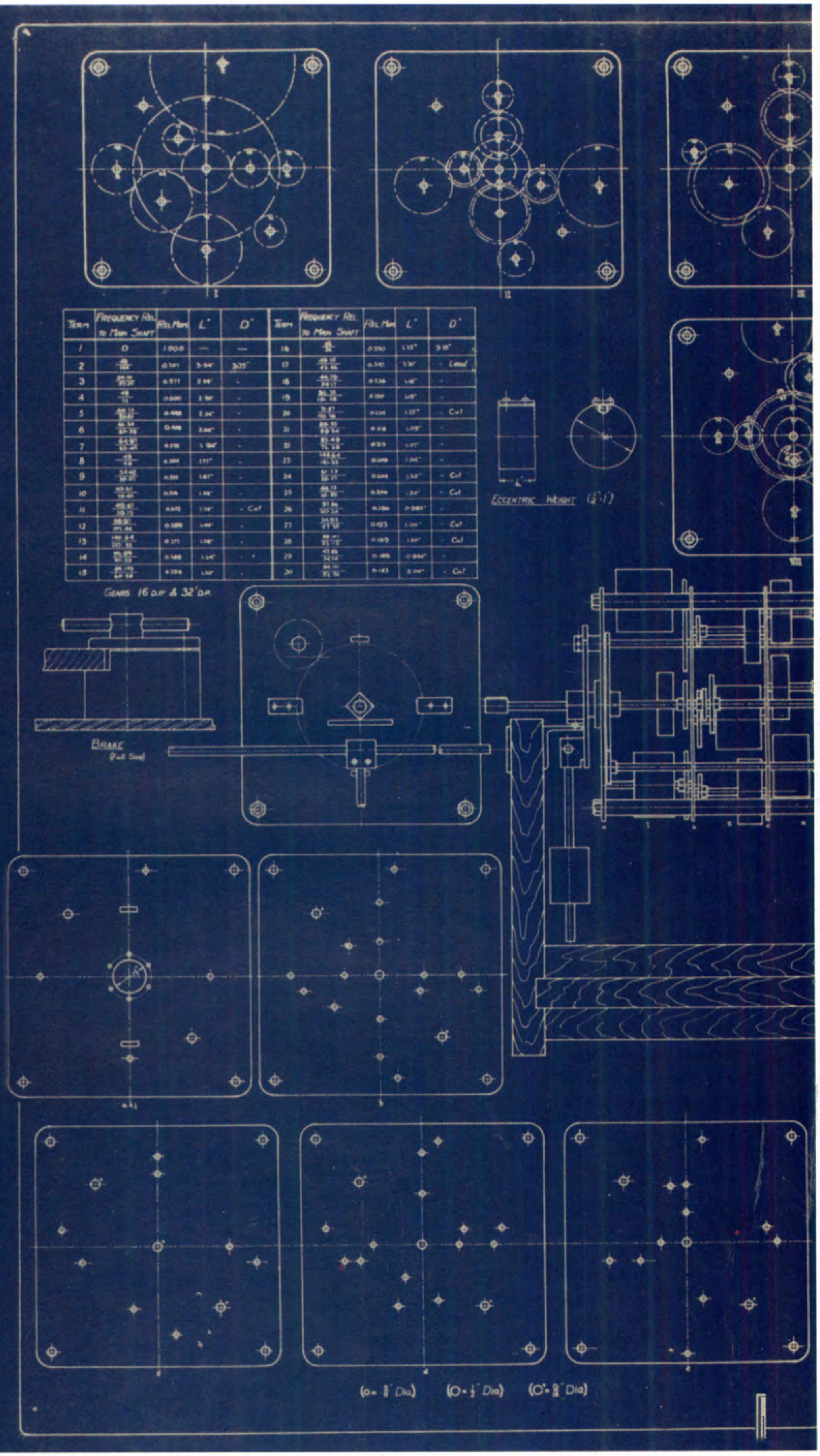
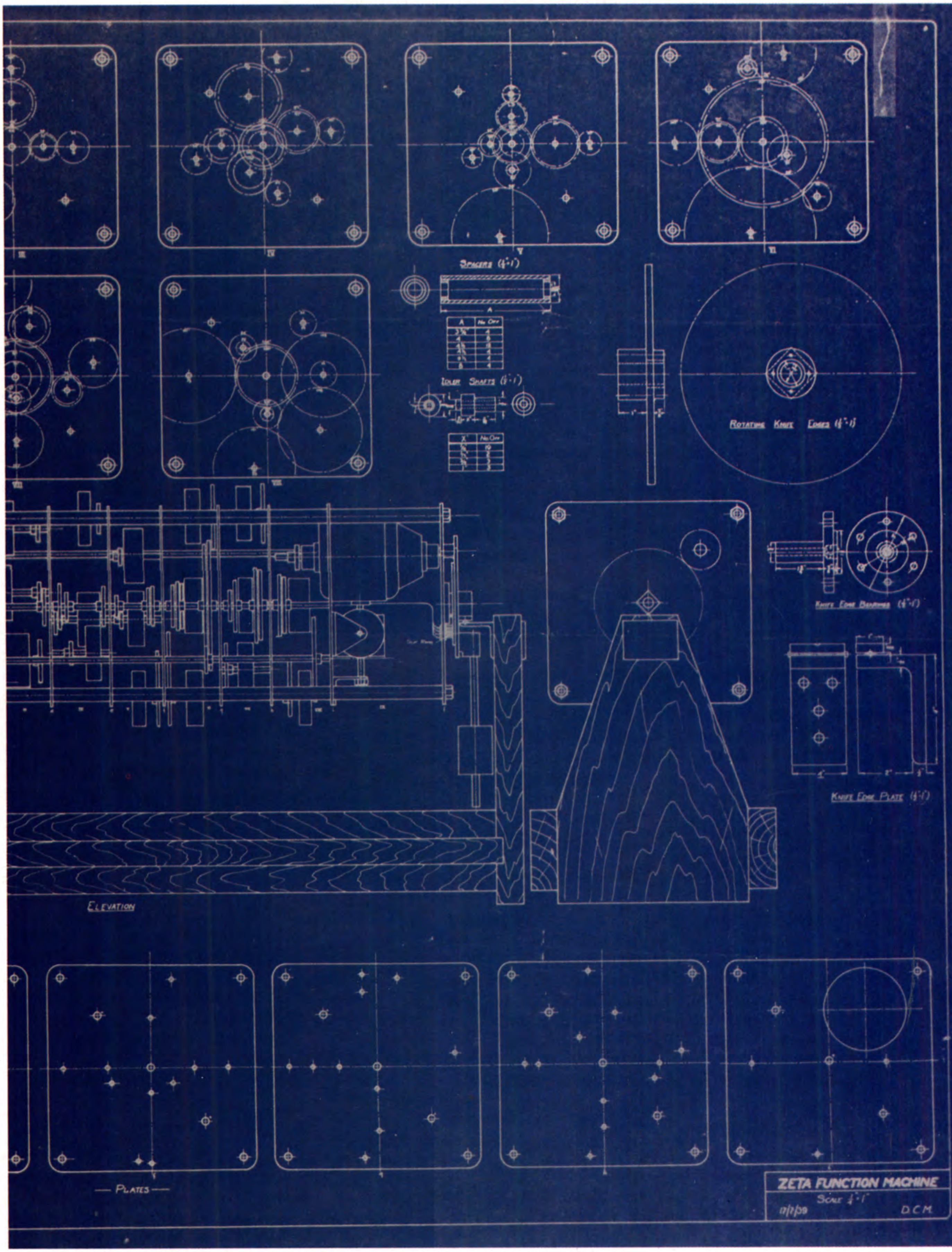
Blueprint for a Turing Machine




ELEVATION
PLATES
ROTATING KNIFE EDGES
ZETA FUNCTION MACHINE
D.C.M.

About the Cover ... and a Bit More

Alan Turing invented the abstract Turing machine in the mid-1930s, one presumably capable of performing—albeit slowly—any calculation that could be performed by anyone at any time. But a bit later in the same decade he designed two rather more physical machines, ones that were by no means universal. The first was a project of his while he was in Princeton, in 1937, and was assembled from relays that he built himself in the workshop of the Princeton physics department. This was to some extent a collaboration with a Canadian graduate student in physics named Malcolm MacPhail, who wrote in a letter to Turing's biographer, Andrew Hodges, "my small contribution to the project was to lend Turing the key to the shop, which was probably against all the regulations, and to show him how to use the lathe, drill press, (etc.) without chopping off his fingers. And so, he wound the relays; and to our surprise and delight the calculator worked." The machine apparently did only one task—it multiplied two integers in binary format, which at that time, when adding machines all used decimal format, would have been an innovation.

The second machine was the one whose blueprint is on the cover and the preceding pages. Its purpose was to assist in verifying the Riemann hypothesis. It was designed in 1939, when Turing was back in Cambridge, with the assistance of Malcolm MacPhail's brother Donald, then a graduate student in engineering. Turing applied for and received a small but helpful grant from the Royal Society to cover the cost of construction, and on the application form he admitted that "Apparatus would be of little permanent value". It was necessarily inaccurate, but the idea was that it would give likely locations for zeroes which could then be checked by more traditional methods.

The Mathematics

From Turing's application to the Royal Society:

It is proposed to make calculations of the Riemann zeta-function on the critical line for $1,450 < t < 6,000$ *with a view to discovering whether all the zeros of the function in this range of* t *lie on the critical line. An investigation for* $0 < t < 1,464$ *has already been made by Titchmarsh. The most laborious part of such calculations consists in the evaluation of certain trigonometrical sums*

$$\sum_{r=1}^{m} \frac{1}{\sqrt{r}} \cos(t \log r - \vartheta) \quad m = \left\lfloor \sqrt{\frac{t}{2\pi}} \right\rfloor$$

In the present calculation it is intended to evaluate these sums approximately in most cases by the use of apparatus somewhat similar to what is used for tide prediction. When this method does not give sufficient accuracy it will be necessary to revert to the straightforward calculation of the trigonometric sums, but this should be only rarely necessary. I am hoping that the use of the tide-predicting machine will reduce the amount of such calculation necessary in a ratio of $50:1$ *or better. It will not be feasible to use already existing tide predictors because the frequencies occurring in the tide problems are entirely different from those occurring in the zeta-function problem. I shall be working in collaboration with D. C. MacPhail, a research student who is an engineer. We propose to do most of the machine-shop work ourselves, and are therefore applying only for the cost of materials, and some preliminary computation.*

The formula referred to is that of Riemann and Siegel, a recent discovery. The function ϑ is defined by the formula

$$\vartheta(t) = \arg \gamma(1/2 + it), \gamma(s) = \pi^{-s/2}\Gamma(s/2).$$

Because of Stirling's formula, it has a simple approximation in the range for large values of t.

In the table on the blueprint, r is in the first column, $\log_8 n$ in the second but expressed as a ratio of products of integers. For example, $\log_8 16 = 4/3 = 48/36$. There seem to be two errors in the table, for $n = 24$ and 30. In the third column is $1/\sqrt{n}$. I do not know what the fourth column L'' is.

Gears

In a letter to Turing's mother after his death, Malcom MacPhail wrote, "Alan's zeta function computer was a device for adding up a large number of sines and cosines of various periods and amplitudes... The gears, of which there were to be

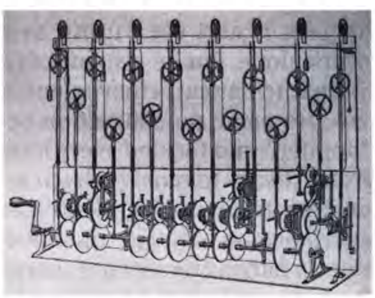
Term	Frequency Rel. to Main Shaft	Rel. Mom	L"	D"
16	48 / 36	0·250	1·25"	3·75"
17	49·53 / 43·46	0.241	1·70"	- Lead
18	44·29 / 34·37	0·236	1·18"	-
19	31·33 / 151·48	0·230	1·15"	-
20	71·67 / 133·76	0·224	1·25"	- Cut
21	89·33 / 59·34	0·218	1·09"	-
22	83·49 / 71·38	0·213	1·07"	-
23	144·64 / 191·32	0·208	1·04"	-
24	61·23 / 25·37	0·204	1·20"	- Cut
25	44·73 / 25·83	0·200	1·20"	- Cut
26	97·88 / 337·34	0·196	0·985"	-
27	34·93 / 57·38	0·193	1·00"	- Cut
28	88·107 / 52·113	0·189	1·20"	- Cut
29	47·82 / 34·70	0·186	0·936"	-
30	64·141 / 52·32	0·183	2·00"	- Cut

Part of the table of gear specs.

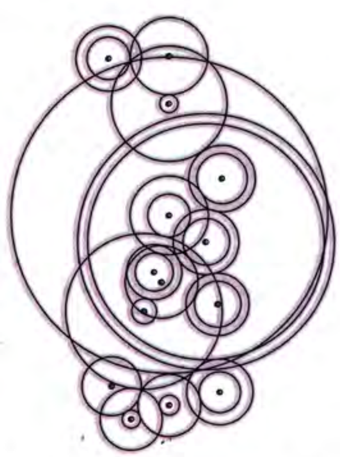
hundreds, were to provide an approximation to the required periods... Alan obtained rational approximations to [the periods]... by the method of continued fractions. A pair of gears having the number of teeth indicated respectively by the numerator and denominator of the fraction would then rotate at speeds having approximately the desired ratio."

The tide-predicting machines referred to by Turing in his application were part of a family designed much earlier by Lord Kelvin. It seems that he had seen one then still in use in Liverpool.

In correspondence with Andrew Hodges, Donald MacPhail mentioned something about a variation in the tide-predicting scheme involving balanced weights, but exactly how this would work is not apparent. The idea of using fractions to approximate real numbers in calculations involving periodic functions, and of using gears to sum such functions is very old. The blueprint doesn't look so much like one of Kelvin's machines as it does the design of the oldest extant computing device, the Antikythera mechanism (from the first century B.C.), a kind of orrery.



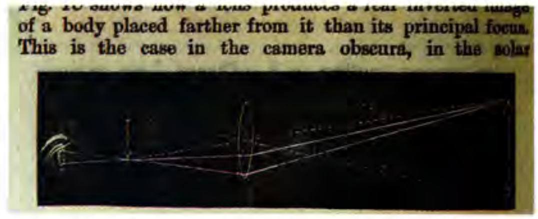
One of Kelvin's tide-predicting machines.



A schematic drawing of the Antikythera mechanism.

Blueprints

Blueprints have disappeared in my lifetime, although their memory (and the word itself in the name of many modern reprographic companies) lingers on. They were invented by the astronomer Sir John Herschel (son of the man who discovered Uranus) around 1840. Paper is coated with a solution, the parts which are to remain white are covered, and subsequent exposure to light turns the uncovered parts a dark, rather attractive, shade of blue ("Prussian" blue presumably because the solution is a compound of cyanide or Prussic acid). They are stable, accurate, and reproducible. They were used occasionally for very high quality book production in the nineteenth century, but I am not aware that the process was ever used for mathematical drawing. About the middle of the nineteenth century, however, traditional wood cuts were frequently replaced by ones with an inverted coloring scheme, and I am tempted to think this was partly influenced by blueprints.



of a body placed farther from it than its principal focus. This is the case in the camera obscura, in the solar

From the article on Light in the 9th edition of the *Encyclopaedia Britannica*.

References

- The blueprint is AMT/C/2 in the Turing Archive of King's College, Cambridge, and the photograph we used was made by the Imaging Services of the Cambridge University Library. Copyright of the image is owned by P. N. Furbank, whom we wish to thank for permission to publish it. Professor Furbank was a friend of Turing as well as executor of his will, and is also the author of many books on literary topics. We also wish to thank Patricia McGuire, archivist of King's College, for much help in assembling material.
- Much of Chapter 4 of Andrew Hodges' definitive biography *The Enigma*, discusses Turing's machines. Hodges gave me much help in writing this, and in particular sent me scans of Turing's application for the Royal Society grant.
- The letter from Malcolm MacPhail to Mrs. Turing is AMT/A/21.
- More information on tide-predicting machines and the Antikythera mechanism can be found in the archives of the AMS Feature Column at `http://www.ams.org/featurecolumn/archive/index.html`.
- The letter from Titchmarsh to Turing is AMT/D/5 in the King's College archive.
- There is much information on blueprints on the Web. A good place to start (naturally) is `http://en.wikipedia.org/wiki/Blueprint`.
- A short obituary of D. C. MacPhail can be found at `http://www.homebiz.ca/News/Archives/011700.htm`.

—Bill Casselman, Graphics Editor
(notices-covers@ams.org)

Book Review

The Essential Turing

Reviewed by Andrew Hodges

The Essential Turing
A. M. Turing, B. Jack Copeland, ed.
Oxford University Press, 2004
US$29.95, 662 pages
ISBN 0198250800

The Essential Turing is a selection of writings of the British mathematician Alan M. Turing (1912–1954). Extended introductions and annotations are added by the editor, the New Zealand philosopher B. Jack Copeland, and there are some supplementary papers and commentaries by other authors. Alan Turing was the founder of the theory of computability, with his paper "On Computable numbers, with an application to the *Entscheidungsproblem*" (Turing 1936). This, a classic breakthrough of twentieth century mathematics, was written when he was twenty-three. In the course of this work in mathematical logic, he defined the concept of the universal machine. As he himself put it, digital computers are practical versions of this concept; and he himself created an original detailed design for an electronic computer in 1945–46. His 1936 analysis of mental rule-based operations was the starting point for his later advocacy of what is now called Artificial Intelligence. His paper "Computing machinery and intelligence" (Turing 1950a) is one of the most cited in modern philosophical literature. His paper in mathematical biology (Turing 1952) then inaugurated a new field in nonlinear applied mathematics. But he was also the leading scientific figure in the British codebreaking effort of the Second World War, with particular responsibility for the German Enigma-enciphered naval communications, though this work remained secret until the 1970s and only in the 1990s were documents from the time made public.

Andrew Hodges is lecturer in mathematics at Wadham College, University of Oxford. His email address is andrew.hodges@wadh.ox.ac.uk.

Many mathematicians would see Turing as a hero for the 1936 work alone. But he also makes a striking exemplar of mathematical achievement in his breadth of attack. He made no distinction between "pure" and "applied" and tackled every kind of problem from group theory to biology, from arguing with Wittgenstein to analysing electronic component characteristics—a strategy diametrically opposite to today's narrow research training. The fact that few had ever heard of him when he died mysteriously in 1954, and that his work in defeating Nazi Germany remained unknown for so long, typifies the unsung creative power of mathematics which the public—indeed our own students and our colleagues in the sciences—should understand much better.

Many therefore will welcome this new edition and the increased availability of Turing's work. But the foregoing remarks should make it clear that defining the Turing *oeuvre* is not straightforward. There is no default option of reproducing published papers and compiling them under a new cover. There is a spectrum ranging from formal publication to reports, talks, unpublished papers, unfinished work, letters, and several areas where other people developed work that he had inspired. Choices here are not easy. Nor it is straightforward to define a genre or field in which to place his work, and the usual criteria of important papers in leading journals are of no use. Turing ignored conventional classifications, and created work which would now be described as at the foundations of computer science or the cognitive sciences, areas which in his time had no clear names of their own.

On top of this, there is the difficulty that his war work was never written for publication and exists only in operational reports which have been released in a chaotic fashion. Indeed, even since this collection was published, Turing's report on his advanced electronic speech scrambler, with the wonderful date 6 June 1944, has emerged (Turing 1944).

In the last decade the World Wide Web has transformed access to original Turing documents. In particular, the collection of Turing's papers at King's College, Cambridge University, is now accessible at `http://www.turingarchive.org`. Those interested in original material are now much less dependent on the work of editors and publishers. Even so, a printed source-book will be valued by many to whom Turing is a somewhat legendary figure, often cited but not easy to look up and quote. Such a book has enormous potential to educate and to inspire.

Copeland's Anthology

We now come to Copeland's own editorial choices. This review will consider points of interest in an order roughly corresponding to the order of Copeland's anthology, which in turn reflects Turing's chronology. But one point should be made clear at the outset. Oxford University Press bills this as "the first purchasable book by Turing", but Copeland's volume is not the first edition of Turing's papers and not the most complete. A four-volume *Collected Works of A. M. Turing* was published by Elsevier (Turing 1992, 2001). This work, totalling some 938 pages, resulted from the protracted collaboration of distinguished mathematicians and computer scientists. Particularly notable is the volume where John L. Britton, as editor, annotated Turing's pure-mathematical work in line-by-line detail.

Unfortunately, little effort was made to promote the *Collected Works*, and the high price guaranteed it few sales, even to university libraries. For this reason, Copeland's new collection, offered at a paperback price, makes a good part of Turing's work much more accessible in practice. It is still odd, however, that Copeland virtually ignores the *Collected Works*. Indeed, an inattentive reader, missing the small print on pages 409, 510, and 581, would remain ignorant of it.

This omission is compounded by another: Copeland does not list Turing's works, so the reader cannot even guess how complete his selection is. There are quotations from some of Turing's papers which have not been included in the work, but there is no overview of Turing's output, nor any explanation of what is considered "essential" and why. Yet it would have been simple for Copeland to explain the relationship of his collection to the existing edition, and to refer readers to it for other (presumably "inessential") Turing works. Indeed, Copeland could justly claim to have advanced upon the *Collected Works* in some areas—as he does when detailing transcription errors in (Turing 1948a). He has included more technical Enigma material than the *Collected Works* did, as well as Turing's late talks on machine intelligence, which Copeland finds particularly important. But the inclusions and exclusions are nowhere systematically listed.

Copeland's edition has all the papers reset in a uniform typography: some readers will always prefer to see the original format and this decision means the loss of original page references. But Copeland is certainly no slouch when it comes to textual detail. For example, he devotes nearly a page to discussion of the spelling of the word "program".

More problematic is the central question of what is "essential". To illustrate how differently the "essential" Turing may appear in different eyes, it is worth recalling the survey of the topologist M. H. A. (Max) Newman, written for the Biographical Memoirs of the Royal Society after Turing's death (Newman 1955). In some ways Turing's mentor and father figure, Newman interestingly defined him as "at heart more of an applied than a pure mathematician" and devoted serious attention only to his mathematical papers. Of course, Turing's war work was then totally secret, but even so Newman's characterization of it as a cruel loss to science was somewhat severe. Computer design and Artificial Intelligence received the briefest of mentions. This was too narrow a mathematical viewpoint, but it did reflect, perhaps, that *sub specie aeternitatis* aspect of mathematics in which Turing shared: he threw himself into the war effort (as did Newman) but never, even in its darkest days, forgot that he was a serious mathematician. In contrast to Newman, Copeland highlights the Enigma cipher machine as the subject of his second main section, the third focal point being Artificial Intelligence and the Turing Test.

Computability and Logic

In one respect, however, Copeland is entirely in unison with Newman, and that is on the topic of computability, which forms the first main section of his volume. *The Essential Turing* includes not only "On Computable Numbers", but also part of a paper by Emil Post which gives some corrections, and another technical commentary by Donald Davies. (For

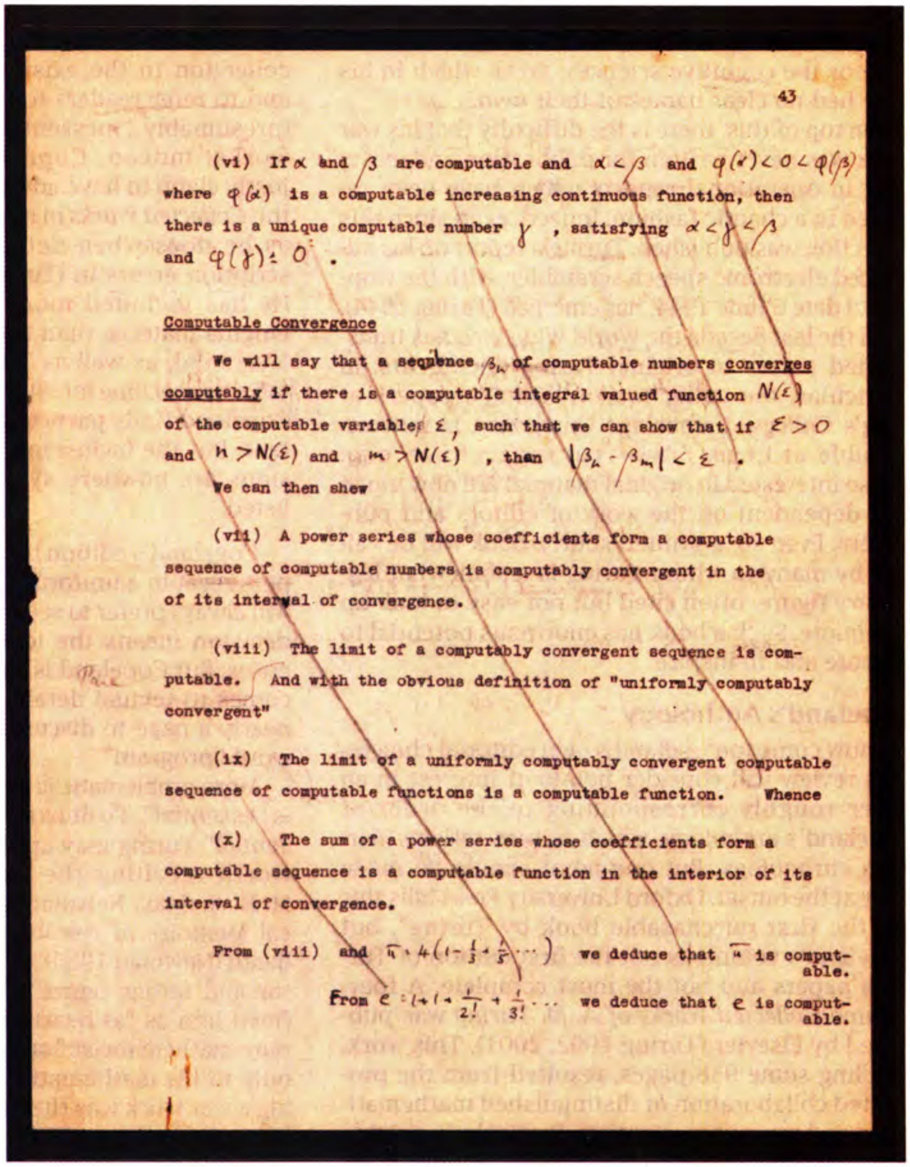
A page from the typescript of Alan Turing's classic 1936 paper, "On computable numbers, with an application to the Entscheidungsproblem".

This text appeared on page 256 of Turing's published paper, with some very minor textual changes. The typewriting is not his, but the inserted mathematical expressions are in his own hand. The lines drawn through the material may mean that it had been retyped, or merely that it was finished with and could be recycled as scrap paper (see below).

The results on this page show that "computable numbers" include all the real numbers that normally arise in mathematics through limit definitions. Although Turing's paper is usually thought of as concerned with the discrete world of mathematical logic, Turing wanted to connect computability with the mainstream of continuous analysis. In fact his opening remarks rather rashly asserted that he would soon give a theory of real functions based on the concept of computable numbers. Turing subsequently abandoned this ambition, leaving it to modern theorists of "computable analysis" to follow up. However, his later note (Turing 1937) made a first step in this direction.

Only six pages of this typescript survive in the Turing Archive at King's College, Cambridge. Their existence has been overlooked because they were used as scrap paper: the reverse sides contain Turing's manuscript for another paper, "A note on normal numbers" (Turing 1936?). This other paper was never published, but there is a modern transcription and detailed annotation in the *Collected Works*. It was probably stimulated by the work of his friend David Champernowne (1933). Champernowne noticed that the number .12345678910111213141 5... is normal in base 10, meaning that its digits and groups of digits are all uniformly distributed in the infinite limit. In attempting to generalize this result, Turing found himself giving constructive definitions of infinite decimals. It seems quite possible that Turing considered this question around 1933–34 and that it influenced the approach he took in 1935 when he formulated his definition of a computable number.

—*Andrew Hodges*



43

(vi) If α and β are computable and $\alpha < \beta$ and $\varphi(\alpha) < 0 < \varphi(\beta)$ where $\varphi(\alpha)$ is a computable increasing continuous function, then there is a unique computable number γ, satisfying $\alpha < \gamma < \beta$ and $\varphi(\gamma) = 0$.

Computable Convergence

We will say that a sequence β_n of computable numbers converges computably if there is a computable integral valued function $N(\varepsilon)$ of the computable variable ε, such that we can show that if $\varepsilon > 0$ and $n > N(\varepsilon)$ and $m > N(\varepsilon)$, then $|\beta_n - \beta_m| < \varepsilon$.

We can then shew

(vii) A power series whose coefficients form a computable sequence of computable numbers is computably convergent in the of its interval of convergence.

(viii) The limit of a computably convergent sequence is computable. And with the obvious definition of "uniformly computably convergent"

(ix) The limit of a uniformly computably convergent computable sequence of computable functions is a computable function. Whence

(x) The sum of a power series whose coefficients form a computable sequence is a computable function in the interior of its interval of convergence.

From (viii) and $\pi = 4(1 - \frac{1}{3} + \frac{1}{5} \cdots)$ we deduce that π is computable.

From $e = 1 + 1 + \frac{1}{2!} + \frac{1}{3!} \cdots$ we deduce that e is computable.

The image above is AMT/C/15/01c.2 at http://www.turingarchive.org/browse.php/C/15, on the website of the Turing Archive of King's College, Cambridge University. Permission to publish has been granted by P. N. Furbank, executor of Turing's will.

References:

D. G. Champernowne (1933), The construction of decimals normal in the scale of ten, *J. Lond. Math. Soc.* **8**, 254-260.

A. M. Turing (1936), On computable numbers, with an application to the Entscheidungsproblem, *Proc. Lond. Math. Soc.* **42**(2), 230-265.

A. M. Turing (1936?), A note on normal numbers, manuscript and typescript available at http://www.turingarchive.org, item C/15. Text in *The Collected Works of A. M. Turing: Pure mathematics*, J. L. Britton (ed.), North-Holland, 1992.

A. M. Turing (1937), On computable numbers, with an application to the Entscheidungsproblem. A correction, *Proc. Lond. Math. Soc.* **43**(2), 544-546.

comparison, Martin Davis's collection of classic papers *The Undecidable* (Davis 1965), recently republished, only offers Post's paper, but offers all of it.) Copeland also gives good didactic material on Hilbert's formalist program and the working of the universal machine. Turing's formidable 1939 paper on ordinal logics (his 1938 Princeton Ph.D. thesis, as supervised by Church) is also included. This issue of the *Notices* includes a review of this paper by Solomon Feferman and a further article by Martin Davis on "Turing Reducibility". In this case it is probably beyond the powers of any editor to make the technical apparatus accessible. Copeland does not try to explain the idea of the lambda-calculus, and does not offer the mathematical content that the authors in the *Notices* supply, but succeeds in giving a clear survey. His treatment is enhanced by the inclusion of some previously unpublished correspondence with Newman from the King's College archive.

Mathematics and Cryptography

After this first section, Turing's mathematical work is marginalised, and the message seems to be that mathematics is less than essential. An example comes in Turing's interest in probability theory. Turing's first substantial research work was an independent proof of the Central Limit Theorem—unmentioned by Copeland, but given an excellent review by Zabell (1995). This won his Fellowship of King's College in 1935. But perhaps more importantly, probability theory was the key to his advanced cryptanalytic methods, which made cryptography into a science. Turing developed new Bayesian inference methods for the Enigma decipherment problem, work in which he was assisted by I. J. (Jack) Good after 1941. Good became a distinguished mathematician and statistician, and his book *Probability and the Weighing of Evidence* (Good 1950) expounded and developed the material that Turing originated but never wrote in his own name. ("Weight of evidence" is essentially equivalent to Shannon's measure of information, which Turing formulated and used independently.) In the *Collected Works*, this work was well accounted for, thanks to Good's wealth of historical material (Good 1992, 1993, 2001). It has inspired modern developments (Orlitsky et al. 2003). Yet the entire subject of probability and statistics is virtually unmentioned in *The Essential Turing*. This is rather like telling the story of the atomic bomb without mentioning nuclear physics. Because of this omission, Copeland does not justify his claim (p. 2) that in *The Essential Turing* "the full story of Turing's involvement in the Enigma is told for the first time."

However, Copeland gives a full description of the Enigma machine and of the early Polish and British methods for deciphering it. The power of Turing's applied logic comes through in his beautiful "simultaneous scanning" method to defeat the plugboard complication of the Enigma. For this, Copeland supplies the pertinent excerpt from Turing's 1940 technical report, as eventually released in 1996, supplemented by a part of A. P. Mahon's more readable internal history of the work. He does not include Mahon's striking conclusion: this gave a mathematician's apology for war work which probably also reflects Turing's sentiments. "While we broke German Naval Cyphers because it was our job to do so and because we believed it to be worthwhile, we also broke them because the problem was an interesting and amusing one. The work of Hut 8 combined to a remarkable extent a sense of urgency and importance with the pleasure of playing an intellectual game."

This omission of Bayesian inference methods also weakens Copeland's claims about the genesis of Artificial Intelligence in wartime Bletchley Park. Copeland argues that the serial trial of a million or so Enigma rotor positions lies behind the identification of "search" in (Turing 1948a) as a concept central to "intelligent machinery". This is an unnecessarily weak link on which to hang the claim. Such brute force "search" was the bluntest of instruments in codebreaking. A more substantial point lies in Turing's successful mechanization of judgment through his quantified "weight of evidence", prefiguring the sophisticated Bayesian inference programs used today in AI applications.

Mathematics and Computer Science

More generally, the hinterland of mathematical theory and practice, as the basis and motivation for advances in computing, is weakly represented. Turing not only worked on computable numbers in the abstract: he knew all about computing numbers in practice. As the *Notices* article by Andrew Booker describes, in 1937–39 Turing developed new methods for investigating the Riemann zeta-function, which led to a need to compute its zeros: for Newman, the abandonment of such work was the cruel blow dealt by the war. But Copeland never mentions complex analysis, nor the special machine Turing designed for computing the zeta-function, nor his 1950 computer program superseding it. It is striking that the first thing Turing did in 1950, when he was able to use one of the world's first computers, was to use it to investigate the zeta-function. (In contrast, he did no experimental work with computers on Artificial Intelligence).

The exclusion of mathematics gives a lopsided view of Turing's mind at work. Thus his pre-war connection with von Neumann through research in continuous groups (Turing 1938), and the development of computability within mathematics (Turing 1950b) go unmentioned. So does Turing's work in the numerical analysis of matrix inversion

(Turing 1948b), although it led the way in showing the viability of numerical methods for large-scale applied mathematical problems and thus made a start on the serious analysis of algorithms. (The work of Higham (1996) has drawn attention to the importance of this work.) The "program proof" of (Turing 1949), anticipating ideas of the 1960s, plays no role: for Copeland the only "essential" topic in computing is Artificial Intelligence.

In defence of this narrow focus it could be said that Turing's central motivation in 1945 did not lie in standard mathematics, nor in practical computer science, but in wanting to build a computer as "a brain". But Turing knew a great deal about the relation between mathematics and the physical world, discreteness and continuity. This knowledge was inseparable from his prospectus for computing and for Artificial Intelligence. Turing's central idea of modelling the brain brought him to consider the approximation to continuous systems by the discrete, including chaotic and thermodynamic effects (Turing 1948a, 1950a). Thus his background as an all-purpose mathematician, rather than as a verbal philosopher, is still important even if this narrow remit is accepted.

Origin of the Digital Computer

Turing's ambition to "build a brain" brings us to the question of his 1945–46 technical proposal for the Automatic Computing Engine (Turing 1946)—the first really detailed electronic computer design and prospectus for what a computer could do. This report was omitted in Newman's 1955 memoir and has had serious recognition only since the 1970s. In this neglect, it stands in complete contrast with the June 1945 "Draft report on the EDVAC" by von Neumann which has always been regarded as the *fons et origo* of the computer. Indeed histories of computers too often tell a story of engineered machines and American corporate history, from Hollerith to microprocessors, without any references to Turing at all. Copeland has previously done much to advance Turing's claim (and British-based work generally), and *The Essential Turing* is billed on its cover as giving "The ideas that gave birth to the computer age." It is therefore odd that the ACE report, the first detailed prospectus for an electronic computer, is omitted. Various bits are quoted, but they do not allow the reader to judge Turing's total vision.

Interdisciplinary culture clashes abound in the question of the origin of the computer. Some computer historians consider the use of electronic components to be the crucial innovation. Binary number representation is often held to be a breakthrough, and I have been surprised to hear Martin Campbell-Kelly, a leading figure in this field, suggest that Turing needed to learn this idea from von Neumann. Copeland, in contrast, focuses very clearly on the principle of the universal machine as the crucial factor. This is unlikely to change the minds of those who consider other issues to be paramount, but it is consistent with Turing's approach.

To argue that Turing's logical work was critical to the modern digital computer, Copeland discusses the concept of stored-program computers in his introduction to (Turing 1936). In fact he virtually *identifies* the concept with the universal machine, by giving Turing's own later account of the connection (Turing 1947). The danger here is that of being ahistorical: that is, forgetting that the 1945 now far in the past was in 1936 far in the future. But Copeland certainly makes a strong argument that von Neumann knew of Turing's ideas and used them, without citation, in the EDVAC report. This is a difficult topic; although von Neumann certainly spoke clearly after 1945 of the importance of Turing's universal machine, there is little to document its influence in the formative period. His strongest evidence remains the statement Brian Randell got from Stanley Frankel long ago (Randell 1972): that "in or about 1943 or '44 von Neumann was well aware of the fundamental importance of Turing's paper of 1936..." More recently Martin Davis has also tackled this problem in his book *The Universal Computer: The Road from Leibniz to Turing* (Davis 2000), giving a vivid discussion of von Neumann's debt to Turing. This is the judgment of a unique source who was immersed in the logic and computer worlds of that early period. Copeland weakens the case for Turing's influence by making no reference to Davis's analysis.

The Essential Turing also makes another argument about the origin of the digital computer, concerned with the question of assigning credit for the Manchester machine which, though tiny, was the world's first working stored program computer in June 1948. But a general problem in both discussions of origins is that Turing never gave a full analysis of his own contribution—in particular, the insight that a program is itself a form of data and can be treated by the computer as such. On the first page of his 1945–46 plan, Turing said that control of an entire calculation could be "looked after by the machine itself". Into this phrase we may read the future of subroutines, languages, compilers, and operating systems; but Turing himself did not spell out those implications systematically. The word "itself" is a Gödelian self-reference; it comes from thinking of programs as data for other programs, but Turing never enlarged his observations on this logical ancestry. Turing used program modification immediately—he described how one could "pretend that the instructions were really numbers"—but did not take the opportunity to explain that such "pretence" embodied an essential and novel principle. Even when making much of the potential of program modification for "learning", he never gave a serious analysis of the program-as-data aspect. Later, while at Manchester, Turing (1950a) wrote that Babbage had "all the essential ideas", thus undermining the appreciation of program-as-data as a crucial advance.

Unfortunately, Turing never gave more than a brief indication of how his 1936 theory led to his 1945–46 design. Secrecy would have inhibited a full account, though Good's example in publishing so much probability theory shows that there was no total bar on the exposition of general principles. However, such a contribution from Turing was hardly encouraged by the embryonic computer industry. In 1953 a semi-popular book *Faster than Thought* (Bowden 1953) surveyed British computing, with Babbage as the star. In contrast, the editor included a philistine "glossary" entry on Turing thus:

> **Türing machine:** In 1936 Dr Turing wrote a paper on the design and limitations of computing machines. For this reason they are sometimes known by his name. The umlaut is an unearned and undesirable addition, due, presumably to an impression that anything so incomprehensible must be Teutonic.

It is not difficult to decode the word "incomprehensible" as meaning "mathematical". A more common view is that Turing's contribution is comprehensible, but purely theoretical. Students often gain the impression that Turing was never connected with anything as vulgar as an actual computer. This is the reverse of the truth: Turing avidly desired the practical business of design and construction. In fact rather than let his claim depend solely upon the abstract principle of 1936, it would be better to emphasise that from 1943 onwards he was in effective command of every aspect of making that principle into a practical proposition—scientific, technological, organizational, motivational. It was in that grasp of its potential that he was the inventor of the computer: that was the essential Turing. In particular his vision for the future of software engineering, based on his deeper understanding of the universal machine being able to "look after itself", was ahead of von Neumann's. Without seeing Turing's ACE report, readers cannot judge his place in the history of the "practical" universal machine. A related point is that Turing's practical wartime experience with digital machinery was crucial, and it would have been worth including some documentation of this experience from his wartime reports. Copeland marginalises these questions, because his attention is concentrated on Artificial Intelligence.

Artificial Intelligence

Copeland is right to emphasise that well before 1956, when the Dartmouth conference inaugurated "Artificial Intelligence" as a research area, Turing had developed strong lines of research, both top down and bottom up, in modern parlance. (Nor was Turing alone in the British scene—he was one of a very lively group of "cybernetic" pioneers.) But as in other ways, Turing suffered the consequences of his own self-effacing reticence. Turing never published the neuron-inspired networks he sketched in (Turing 1948a), even though they played a role in motivating the arguments in his famous 1950 paper about the possibility of learning machines. Nor, oddly, did he try them out when the fully engineered Manchester computer became available. Here another surprising omission comes in Copeland's discussion. Copeland and Proudfoot (1996) drew fresh attention to Turing's 1948 neural architectures (which had been published in 1968 and 1969, and then in the *Collected Works* in 1992). A young computer scientist, Christof Teuscher, then did Ph.D. work in implementing and exploring them; this has won several awards, including one from the European Research Consortium for Informatics and Mathematics. His publications (Teuscher 2002, 2004) are not referenced by Copeland.

Mathematics and Biology

Of Turing's theory of morphogenesis, only the published work (Turing 1952) is included by Copeland. Turing left much more work unfinished at his death. The editor of the *Morphogenesis* volume of the *Collected Works*, P. T. Saunders, edited and included the most coherent parts of these manuscripts. Copeland does not include any of this. His footnote does make a rare concession to the existence of the *Collected Works*, and he does cite the much more extensive work by Jonathan Swinton (2004), which appeared in another volume of Turing-inspired studies, *Alan Turing: Life and Legacy of a Great Thinker*, edited by Christof Teuscher (2004). But Copeland gives no hint of the content of these other scholars' work, preferring to concentrate on his own interpretation of the material in (Turing 1952).

That interpretation is, unfortunately, skewed by Copeland's insistence on describing Turing's theory as "Artificial Life", a term coined in the 1980s, and his linking of it to genetic and evolutionary algorithms. But these "a-life" developments are much closer to von Neumann's ideas; Turing's work in mathematical biology was essentially complementary. Genetic algorithms explore the logic of evolution without the constraints of physical embodiment. Turing's work attacked the question of what paths could be physically available for evolution to exploit. It was rooted in physical chemistry and used techniques in nonlinear differential equations, entirely different from discrete logic. Swinton has called it "good old-fashioned applied mathematics". There is a slender connection with Turing's machine-intelligence ideas, through the question of brain growth, but his morphogenetic theory really forms quite a different field of enquiry.

Philosophy of Computation

The absence of Turing's unpublished work in mathematical biology may disappoint some readers, but applied mathematics is not Copeland's strength, and it is not surprising that it is somewhat sidelined. His emphasis is naturally on the philosophy of computation, and this is the area where one would expect the greatest expertise. This is, indeed, the topic emphasised by Copeland in his discussion of Turing's late writing on AI, which forms his main claim to new scholarship. In particular, a topic central to Copeland's approach is the discussion of *Church's thesis* concerning the definition of "effectively calculable". Turing entered into this subject in a somewhat awkward way: in 1935 Church proposed a definition of effective calculability in terms of the lambda-calculus. Turing's 1936 definition of computability turned out to be mathematically equivalent, and he had to write an appendix to his paper showing this, so delaying his publication. But Church in turn accepted that Turing's analysis of computation gave a far more direct and intuitive argument for why this definition should be made. It is common now to refer to this joint position as the *Church-Turing thesis*.

In many earlier articles, e.g., (Copeland 2000, 2002), Copeland has made very distinctive claims about the Church-Turing thesis. Surprisingly, he has not made these claims so prominently in *The Essential Turing*: it is more that they lurk behind the prefaces and annotations. But they deserve review here nevertheless: it is important for mathematicians to be aware of what philosophers are making of their work, and students of this volume should be aware that the "Further Reading" recommended by its editor may lead them to highly questionable statements.

The main point is that nowadays two different versions of the Church-Turing thesis can be stated, concerning what could be done by (1) a *human being* carrying out a process mechanically, or (2) *any physical process*. It is certainly of interest to study Turing's texts in the light of this modern framework. But it should be borne in mind that even the word "Thesis" was not used until 1952, and that the "physical" Church-Turing thesis was not clearly distinguished and examined until about 1980.

Copeland's distinctive contribution has been his insistence that Turing and Church were always crystal clear that their ideas were absolutely restricted to the model of the human being working to rule. Moreover according to Copeland (2002), the reason for this restriction was specifically that there might be more general machines capable of computing functions which the human worker could not compute—that is to say, functions which according to Turing's definition, we call uncomputable.

In the same article, Copeland asserts Church's agreement with Turing on this question. But in the relevant text of (Church 1937), one finds that Church did *not* actually characterise computable functions as those which can be produced by the human worker; instead he defined computability in terms of "a computing machine, occupying a finite space and with working parts of finite size", describing the human worker as a particular case. In a recent article Copeland (2006) has argued that this is because Church's term "computing machine" means, by definition, a machine designed to imitate human work—whereas the term "machine", *tout court*, would imply something *not* thus restricted. But a glance shows that these writers were unaware of this verbal distinction. Thus Church, in his review of Post's work, restated the definition in terms of an "arbitrary machine". Turing, in the opening statement of his 1936 paper, said that "a number is computable if its decimal can be written down by a machine." In the formal statement in (Turing 1939), Turing characterized effective calculability in terms of what "could be carried out by a machine"—without mentioning the human model at all.

Turing later spoke of human rule-followers, mechanical processes, and physical machines without drawing any attention to the distinction Copeland insists upon as essential. Turing did indeed often explain the scope of a computer in terms of replacing the work of a human calculator, but he also said that a universal machine could replace the "engineering" of special-purpose machines. Turing's post-war lecture to mathematicians (Turing 1947) opened by saying he had been led to the universal machine by analysing "digital computing machines". (He continued by comparing the digital computer *favourably* with differential analysers, showing that he did not see its digital character as a real restriction on its scope.) Turing's post-war focus was in what he called "man as a machine", and he was naturally drawn to the picture of the brain as a physical machine.

This is where this question starts to become interesting, because it is bound up closely with arguments for and against the possibility of Artificial Intelligence. Turing's famous paper (Turing 1950a), appealed to the idea that the brain, as a physical machine, could be simulated by a computer. It was implicit in his estimate of the number of bits of storage in the brain, and it was addressed directly in what Turing called the Argument from Continuity of the Nervous System. In the later radio talk (Turing 1951), he explained this idea even more explicitly, stating the idea that a universal machine could do the work "of any machine into which one can feed data and which will later print out results." So in *The Essential Turing*, Copeland steps in (p. 479) to inform the reader that this was *not the Church-Turing thesis* but a *different* thesis. In modern terms, one would indeed make a distinction, as explained above. But in 1951 there was no well-defined "thesis" at all, and this distinction did not exist.

This would be little more than a quibble over words and definitions, if it were not for the fact that Copeland claims to have made a discovery in Turing's texts, overlooked by everyone else, which presages a revolution in science and technology. As already mentioned, Copeland holds that Turing always had clearly in mind that there could be physical machines ("hypermachines") with the ability to compute uncomputable functions. In fact, Copeland specifically identifies the "oracle-machines" of (Turing 1939) as being just such entities.

What are these "oracle-machines"? The following comments should be read in conjunction with the articles by Solomon Feferman and Martin Davis. Turing machines typically solve infinitely many cases of a problem. For instance, there is a Turing machine which given any integer n, correctly decides whether n is prime. But Turing (1936) showed the existence of well-defined problems where no Turing machine can solve all the cases. Nowadays perhaps the best known such problem is Hilbert's Tenth Problem, that of deciding whether a Diophantine equation has a solution. A Diophantine oracle would have the property that given any Diophantine equation (e.g., the Fermat-Wiles equation) it would supply the truth about its solubility. Mathematicians would naturally see oracles as a purely mathematical definition, useful for defining *relative* computability: if you could solve Hilbert's Tenth Problem, what else could you do? This became a standard idea in the text of (Davis 1958). In fact Turing (1939) did indeed use the oracle in this way, but he also had an extra-mathematical interpretation for it: he saw the oracle as related to what he called "intuition", the nonmechanical step involved in seeing the truth of a formally unprovable Gödel sentence. However he made no suggestion of engineering any such object, and emphasised that an oracle, by its nature, could not be a machine.

In contrast, Copeland claimed in a *Scientific American* article (Copeland and Proudfoot 1999) that "Turing did imagine" an oracle which would physically "work", e.g., by measuring "a quantity of electricity" to infinite precision, and that now "the search is under way" for such oracles. These, if found, would bring about a new revolution in computing. In another paper (Copeland 1998) he asserted that the oracles were only theoretical for Turing in the same sense as the atomic Turing machine component operations were theoretical. This is also a far-fetched claim: the primitive operations of a Turing machine could be implemented by simple switches such as 1936 automatic telephone exchanges already used. Oracles need to store an infinite database.

Indeed Copeland is determined to detect references to physical oracles in Turing's later work. He has two main arguments, both fallacious. In the first (Copeland 2000, 2006), he identifies the "infinite store" appearing in the semi-popular account of computability in (Turing 1950) as a reference to the infinite database of an oracle. It is not: Turing's analysis makes it obvious that this is simply a description of the unlimited tape available to a Turing machine. This passage explains computability as a theoretical bound on what actual finite computers can do; in fact it emphasises the *finiteness* of the means Turing believed necessary for the simulation of human intelligence—not much more than a trillion bits of storage. It is surprising that Copeland should insist on this quite elementary misreading, given that he has—for well over a decade—devoted so much scholarship to Turing's work.

The second argument, more subtle and complex, involves computability, randomness, and learning. It is most clearly stated in (Copeland 2006), which holds that the pre-war oracles reappear as a necessary feature in Turing's post-war theory of machine-based learning. Turing's 1948 work had a picture of neural nets which could be trained into functionality by "reward" and "punishment" operations—the same fundamental scenario as in modern bottom-up Artificial Intelligence techniques. Copeland holds that this model of learning is a *development* of the idea of intuition. To this it may be objected that Turing's post-war behaviorist model is pretty well the *antithesis* of his prewar picture of "intuitive" knowledge. But leaving this general question aside, Copeland's proposal has the more concrete problem that a program modified in accordance with some finite learning or training process, is only modified into a different program. It does not go beyond the scope of computable functions. Nor is there any reason to suppose, from Turing's writing, that the process of finding better and better algorithms requires access to an uncomputable source. Yet Copeland (2006) concludes by describing a specific procedure in which an oracle, as defined in (Turing 1939), would supply the training sequence. This is quite foreign to Turing's exposition: not only is such an oracle essentially infinite, but it holds exact data and is nothing like the random trial-and-error process Turing suggested, using human infancy as a model.

One might add that this terminology of "hypercomputing", with its spurious connotation of technological feasibility, has been too readily allowed to pass into the currency of computer science. The generally valuable Turing-inspired volume edited by Teuscher (2004), contains not only Copeland's views on this subject but another article (by M. Stannett) advocating absurd propositions such as that the Fourier decomposition of a function implies the possibility of infinite data storage in a finite piece of wire.

Turing did not use the word "machine" with perfect clarity, and it is impossible to read past minds. What we can see, however, is the *mathematical and scientific use* to which he put his words. And Copeland's insistence on detecting implemented oracles between the lines of Turing's post-war texts renders Turing's advocacy of Artificial Intelligence incoherent. Why should Turing have devoted so much time and trouble to promulgating his "heretical" theory that intelligence could be simulated by a computer program if, all the time, he envisaged the engineering of physical processes beyond the scope of digital computers, or considered it vital to have access to an uncomputable oracle? And, if these issues were as crucial as Copeland believes, why did Turing not make them plain rather than leave them in an obscure form requiring decryption by philosophers?

Nevertheless, *The Essential Turing* has the great virtue of making Turing's own texts accessible, so that readers can assess such arguments for themselves. A particular case is that 1951 radio talk, which was unfortunately omitted from the *Collected Works*. Indeed Copeland has usefully drawn attention to an important feature of that talk. When Turing then discussed the computer simulation of the brain, and the idea that a universal machine could simulate any machine, he touched on the possibility that this might actually be impossible, even in principle, because of the uncertainty in quantum mechanics. This clearly departed from what he had said in (Turing 1950a), which made no mention of quantum mechanics when discussing the mechanical simulation of the brain. Turing attributed this view to Eddington, rather than assert it as his own view, but we can see that he did take it seriously as an objection to his Artificial Intelligence thesis.

So Turing did indeed contribute to the long process of distinguishing and discussing a "physical" Church-Turing thesis. With further experience and thought, Turing naturally developed a clearer idea of what he considered involved in "mind" and "machine". He did not, as Copeland implicitly assumes, adopt a philosophical position and hold it unchangingly from start to end. Evolution is in the nature of mathematics and science, and it continues vigorously now. This 1951 development, stating a new objection from quantum mechanics, is especially interesting because it is just the objection to AI which in the 1980s Roger Penrose developed into a full-scale critique by combining it with what Turing called the Mathematical Argument. Copeland does not make this connection, and seems not to notice the significance of Eddington who had played an energising part in Turing's early thought. Elsewhere (p. 477) he quotes from Turing's juvenile but striking essay from about 1932, based on Eddington's ideas about the mind, yet does not point out its reference to quantum mechanical indeterminacy.

Logic and Physics

The trouble here seems to be Copeland's lack of interest in physics, as profound as his lack of concern with statistics and number theory. The *Essential Turing* does not mention Turing's notes and letters describing his last year of interest in fundamental physics. It does not include Robin Gandy's letter to Newman describing Turing's ideas at the time of his death and the only hint we have about what Turing might have done if he had lived (Gandy 1954). This refers to Turing's intent to find a "new quantum mechanics", definitely suggesting he was trying to *defeat* the Eddington (and later Penrose) objection along with the others. He noted with interest the surprising feature of standard quantum mechanics that in the limit of continuous observation a system cannot evolve. This, nowadays known as the "Quantum Zeno effect", was not deep or new but gave a vivid pointer to an area where Turing might have been led had he lived.

John Britton, editing the *Pure Mathematics* volume of the *Collected Works*, contributed a story from personal recollection. Turing gave a talk at Manchester about the number N, which he defined in terms of the probability that a piece of chalk would jump from his hand and write a line of Shakespeare. Probabilities and physical prediction were natural starting points for his mathematical thought. From an early age, Turing was aware of the importance of physical materialism, the magical power of mathematics to encode the laws of physical matter, and the puzzle of the apparent conflict of physical determinism with human will and consciousness. Turing's mathematical life started with Einstein and Eddington, and it ended in the same physical world. Eddington asked how could "this collection of ordinary atoms be a thinking machine?" and Turing found a new answer. The "imitation game" is at heart the drama of materialist scientific explanation for the phenomenon of Mind, with the mathematical discovery of computability as its new leading actor.

The Essential Turing does not present this background, and it shifts the emphasis from mathematics to philosophy, but it makes a good part of Turing accessible to readers of all kinds: his vivid and direct writing will now reach a new audience and encourage new thoughts. We may regret that the self-effacing Turing did not write more on the genesis and development of his theory of minds and machines. We may likewise regret that he did not write more about his extraordinary life and experiences. Commentators will, necessarily, have to interpret those silences, and these interpretations will arouse controversy. But the controversies are always modern, challenging, and as wide-ranging as Alan Turing himself.

References

[1] A. M. Turing (1992, 2001), *The Collected Works of A. M. Turing: Mechanical Intelligence*, (D. C. Ince, ed.), North-Holland, 1992; *Pure Mathematics*, (J. L. Britton, ed.), North-Holland, 1992; *Morphogenesis*, (P. T. Saunders, ed.), North-Holland, 1992; *Mathematical Logic*, (R. O. Gandy and C. E. M. Yates, eds.), North-Holland, 2001. These volumes are referred to below as CW1, 2, 3, 4 respectively.

[2] V. Bowden (ed.) (1953), *Faster than Thought*, Pitman, London, 1953.

[3] A. Church (1937), Review of Turing (1936), *J. Symbolic Logic* **2**, 42–3.

[4] B. J. Copeland and D. Proudfoot (1996), On Alan Turing's anticipation of connectionism, *Synthese* **108**, 361–377.

[5] B. J. Copeland (1998), Turing's o-machines, Penrose, Searle and the brain, *Analysis* **58**, 128–138.

[6] B. J. Copeland and D. Proudfoot (1999), Alan Turing's forgotten ideas in computer science, *Scientific American* **253**:4, 98–103.

[7] B. J. Copeland (2000), Narrow versus wide mechanism: including a re-examination of Turing's views on the mind-machine issue, *J. of Philosophy* **96**, 5–32.

[8] ____ (2002), The Church-Turing thesis, in E. N. Zalta (ed.), *Stanford Encyclopedia of Philosophy*, `http://plato.stanford.edu`.

[9] ____ (2006), Turing's thesis, in *Church's thesis after 70 years*, (A. Olszewski et al., eds.), Ontos Verlag, 2006.

[10] M. Davis (1958), *Computability and Unsolvability*, McGraw-Hill, 1958; new edition Dover, 1982.

[11] ____ (1965), *The Undecidable*, Raven Press, 1965; Dover 2004.

[12] ____ (2000), *The Universal Computer: The Road from Leibniz to Turing*, Norton; new edition as *Engines of Logic*, Norton, 2001.

[13] R. O. Gandy (1954), letter to M. H. A. Newman, available at `http://www.turingarchive.org`, item D/4. Text in CW4.

[14] I. J. Good (1950), *Probability and the Weighing of Evidence*, Griffin, London, 1950.

[15] ____ (1992), Introductory remarks for the article in *Biometrika* **66** (1979), "A. M. Turing's statistical work in World War II", in CW2.

[16] ____ (1993), Enigma and fish, in *Codebreakers*, (F. H. Hinsley and A. Stripp, eds.), Oxford University Press, 1993.

[17] ____ (2001), Commentary on Turing's manuscript "Minimum cost sequential analysis", in CW4.

[18] N. J. Higham (1996), *Accuracy and Stability of Numerical Algorithms*, SIAM, 1996.

[19] M. H. A. Newman (1955), Alan Mathison Turing, *Biographical Memoirs of Fellows of the Royal Society* **1**, 253–263, and in CW4.

[20] A. Orlitsky, N. P. Santhanam, and J. Zhang (2003), Always good Turing: Asymptotically optimal probability estimation, *Science* **302**, 427–431.

[21] B. Randell (1972), On Alan Turing and the origins of digital computers, *Machine Intelligence* **7**.

[22] J. Swinton (2004), Watching the daisies grow: Turing and Fibonacci phyllotaxis, in (Teuscher 2004). See also `http://www.swintons.net/jonathan/turing.htm`.

[23] C. Teuscher (2002), *Turing's Connectionism, an Investigation of Neural Network Architectures*, London: Springer, 2002. See also his article "Turing's Connectionism" in (Teuscher 2004).

[24] ____ (ed.) (2004), *Alan Turing: Life and Legacy of a Great Thinker*, Berlin: Springer, 2004.

[25] A. M. Turing (1936), On computable numbers, with an application to the Entscheidungsproblem, *Proc. Lond. Math. Soc.* **42**(2), 230–265. In CW4 and ET.

[26] ____ (1938), Finite approximations to Lie groups, *Ann. of Math.* **39**(1), 105–111. In CW2.

[27] ____ 1939), Systems of logic based on ordinals, *Proc. Lond. Math. Soc.* **45**(2), 161–228. In CW4 and ET.

[28] ____ (1944), Speech system "Delilah"—report on progress, typescript dated 6 June 1944. National Archives (London), box HW 62/6.

[29] ____ (1946), Proposed electronic calculator, copy of typescript at `http://www.turingarchive.org`, item C/32. Printed version in CW1.

[30] ____ (1947), Lecture to the London Mathematical Society, 20 February 1947, typescript at `http://www.turingarchive.org`, item B/1. Text in CW1 and in ET.

[31] ____ (1948a), Intelligent machinery, National Physical Laboratory report, typescript available at `http://www.turingarchive.org`, item C/11. Text in CW1 and (better) in ET.

[32] ____ (1948b), Rounding-off errors in matrix processes, *Quart. J. Mech. Appl. Math.* **1**, 287–308. In CW1.

[33] ____ (1949), Checking a large routine, paper given at EDSAC inaugural conference, in CW1.

[34] ____ (1950a), Computing machinery and intelligence, *Mind* **59**, 433–460. In CW1 and ET.

[35] ____ (1950b), The word problem in semi-groups with cancellation, *Ann. of Math.* **52**(2) 491–505. In CW2.

[36] ____ (1951), Can digital computers think? BBC radio talk, typescript available at `http://www.turingarchive.org`, item B/5. Text in ET.

[37] ____ (1952), The chemical basis of morphogenesis, *Phil. Trans. R. Soc. Lond.*, **B 237**, 37–72. In CW3 and ET.

[38] S. Zabell, (1995), Alan Turing and the central limit theorem, *American Mathematical Monthly* **102**, 483–494.

Turing's Thesis

Solomon Feferman

In the sole extended break from his life and varied career in England, Alan Turing spent the years 1936–1938 doing graduate work at Princeton University under the direction of Alonzo Church, the doyen of American logicians. Those two years sufficed for him to complete a thesis and obtain the Ph.D. The results of the thesis were published in 1939 under the title "Systems of logic based on ordinals" [23]. That was the first systematic attempt to deal with the natural idea of overcoming the Gödelian incompleteness of formal systems by iterating the adjunction of statements—such as the consistency of the system—that "ought to" have been accepted but were not derivable; in fact these kinds of iterations can be extended into the transfinite. As Turing put it beautifully in his introduction to [23]:

> The well-known theorem of Gödel (1931) shows that every system of logic is in a certain sense incomplete, but at the same time it indicates means whereby from a system L of logic a more complete system L' may be obtained. By repeating the process we get a sequence $L, L_1 = L', L_2 = L'_1$ ... each more complete than the preceding. A logic L_ω may then be constructed in which the provable theorems are the totality of theorems provable with the help of the logics L, L_1, L_2, ... Proceeding in this way we can associate a system of logic with any constructive ordinal. It may be asked whether such a sequence of logics of this kind is complete in the sense that to any problem A there corresponds an ordinal α such that A is solvable by means of the logic L_α.

Using an ingenious argument in pursuit of this aim, Turing obtained a striking yet equivocal partial completeness result that clearly called for further investigation. But he did not continue that himself, and it would be some twenty years before the line of research he inaugurated would be renewed by others. The paper itself received little attention in the interim, though it contained a number of original and stimulating ideas and though Turing's name had by then been well established through his earlier work on the concept of effective computability.

Here, in brief, is the story of what led Turing to Church, what was in his thesis, and what came after, both for him and for the subject.[1]

From Cambridge to Princeton

As an undergraduate at King's College, Cambridge, from 1931 to 1934, Turing was attracted to many parts of mathematics, including mathematical logic.

Solomon Feferman is emeritus professor of mathematics and philosophy at Stanford University. His email address is sf@csli.stanford.edu.

[1] *I have written about this at somewhat greater length in [10]; that material has also been incorporated as an introductory note to Turing's 1939 paper in the volume,* Mathematical Logic *[25] of his collected works. In its biographical part I drew to a considerable extent on Andrew Hodges' superb biography,* Alan Turing: The Enigma *[16].*

In 1935 Turing was elected a fellow of King's College on the basis of a dissertation in probability theory, *On the Gaussian error function*, which contained his independent rediscovery of the central limit theorem. Earlier in that year he began to focus on problems in logic through his attendance in a course on that subject by the topologist M. H. A. (Max) Newman. One of the problems from Newman's course that captured Turing's attention was the *Entscheidungsproblem*, the question whether there exists an effective method to decide, given any well-formed formula of the pure first-order predicate calculus, whether or not it is valid in all possible interpretations (equivalently, whether or not its negation is satisfiable in some interpretation). This had been solved in the affirmative for certain special classes of formulas, but the general problem was still open when Turing began grappling with it. He became convinced that the answer must be negative, but that in order to demonstrate the impossibility of a decision procedure, he would have to give an exact mathematical explanation of what it means to be computable by a strictly mechanical process. He arrived at such an analysis by mid-April 1936 via the idea of what has come to be called a *Turing machine*, namely an idealized computational device following a finite table of instructions (in essence, a program) in discrete effective steps without limitation on time or space that might be needed for a computation. Furthermore, he showed that even with such unlimited capacities, the answer to the general *Entscheidungsproblem* must be negative. Turing quickly prepared a draft of his work entitled "On computable numbers, with an application to the *Entscheidungsproblem*"; Newman was at first skeptical of Turing's analysis but then became convinced and encouraged its publication.

Neither Newman nor Turing were aware at that point that there were already two other proposals under serious consideration for analyzing the general concept of effective computability: one by Gödel called *general recursiveness*, building on an idea of Herbrand, and the other by Church, in terms of what he called the *λ-calculus*.[2] In answer to the question, "Which functions of natural numbers are effectively computable?", the Herbrand-Gödel approach was formulated in terms of finite systems of equations from which the values of the functions are to be deduced using some elementary rules of inference; since the functions to be defined can occur on both sides of the equations, this constitutes a general form of recursion. Gödel explained this in lectures on the incompleteness results during his visit to the Princeton Institute for Advanced Study in 1934, lectures that were attended by Church and his students Stephen C. Kleene and J. Barkley Rosser. But Gödel regarded general recursiveness only as a "heuristic principle" and was not himself willing to commit to that proposed analysis. Meanwhile Church had been exploring a different answer to the same question in terms of his λ-calculus—a fragment of a quite general formalism for the foundation of mathematics, whose fundamental notion is that of arbitrary functions rather than arbitrary sets. The "λ" comes from Church's formalism according to which if $t[x]$ is an expression with one or more occurrences of a variable x, then $\lambda x.t[x]$ is supposed to denote a function f whose value $f(s)$ for each s is the result, $t[s/x]$, of substituting s for x throughout t.[3] In the λ-calculus, function application of one expression t to another s as argument is written in the form ts. Combining these, we have the basic evaluation axiom: $(\lambda x.t[x])s = t[s/x]$.

Using a representation of the natural numbers in the λ-calculus, a function f is said to be *λ-definable* if there is an expression t such that for each pair of numerals $\boldsymbol{n}$ and $\boldsymbol{m}$, $t\boldsymbol{n}$ evaluates out to $\boldsymbol{m}$ if and only if $f(n) = m$. In conversations with Gödel, Church proposed λ-definability as the precise explanation of effective computability ("Church's Thesis"), but in Gödel's view that was "thoroughly unsatisfactory". It was only through a chain of equivalences that ended up with Turing's analysis that Gödel later came to accept it, albeit indirectly. The first link in the chain was forged with the proof by Church and Kleene that λ-definability is equivalent to general recursiveness. Thus when Church finally announced his "Thesis" in published form in 1936 [1], it was in terms of the latter. In that paper, Church applied his thesis to demonstrate the effective unsolvability of various mathematical and logical problems, including the decision problem for sufficiently strong formal systems. And then in his follow-up paper [2] submitted April 15, 1936—just around the time Turing was showing Newman *his* draft—Church proved the unsolvability of the *Entscheidungsproblem* for logic. When news of this work reached Cambridge a month later, the initial reaction was great disappointment at being scooped, but it was agreed that Turing's analysis was sufficiently different to still warrant publication. After submitting it for publication toward the end of May 1936, Turing tacked

[2] *The development of ideas about computability in this period by Herbrand, Gödel, Church, Turing, and Post has been much written about and can only be touched on here. For more detail I recommend the article by Kleene [17] and the articles by Hodges, Kleene, Gandy, and Davis in Part I of Herken's collection [15], among others. One of the many good online sources with further links is at* `http://plato.stanford.edu/entries/church-turing/`, *by B. J. Copeland.*

[3] *One must avoid the "collision" of free and bound variables in the process, i.e., no free variable z of s must end up within the scope of a "λz"; this can be done by renaming bound variables as necessary.*

on an appendix in August of that year in which he sketched the proof of equivalence of computability by a machine in his sense with that of λ-definability, thus forging the second link in the chain of equivalences [21].

In Church's 1937 review of Turing's paper, he wrote:

> As a matter of fact, there is involved here the equivalence of three different notions: computability by a Turing machine, general recursiveness in the sense of Herbrand-Gödel-Kleene, and λ-definability in the sense of Kleene and the present reviewer. Of these, the first has the advantage of making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately... The second and third have the advantage of suitability for embodiment in a system of symbolic logic.[4]

Thus was born what is now called the *Church-Turing Thesis*, according to which the effectively computable functions are exactly those computable by a Turing machine.[5] The (Church-)Turing Thesis is of course not to be confused with Turing's thesis under Church, our main subject here.

Turing in Princeton

On Newman's recommendation, Turing decided to spend a year studying with Church, and he applied for one of Princeton's Procter fellowships. In the event, he did not succeed in obtaining it, but even so he thought he could manage on his fellowship funds from King's College of 300 pounds per annum, and so Turing came to Princeton at the end of September 1936. The Princeton mathematics department had already been a leader on the American scene when it was greatly enriched in the early 1930s by the establishment of the Institute for Advanced Study. The two shared Fine Hall until 1940, so that the lines between them were blurred and there was significant interaction. Among the mathematical leading lights that Turing found on his arrival were Einstein, von Neumann, and Weyl at the Institute and Lefschetz in the department; the visitors that year included Courant and Hardy. In logic, he had hoped to find—besides Church—Gödel, Bernays, Kleene, and Rosser. Gödel had indeed commenced a second visit in the fall of 1935 but left after a brief period due to illness; he was not to return until 1939. Bernays (noted as Hilbert's collaborator on his consistency program) had visited 1935–36, but did not visit the States again until after the war. Kleene and Rosser had received their Ph.D.'s by the time Turing arrived and had left to take positions elsewhere. So he was reduced to attending Church's lectures, which he found ponderous and excessively precise; by contrast, Turing's native style was rough-and-ready and prone to minor errors, and it is a question whether Church's example was of any benefit in this respect. They met from time to time, but apparently there were no sparks, since Church was retiring by nature and Turing was somewhat of a loner.

In the spring of 1937, Turing worked up for publication a proof in greater detail of the equivalence of machine computability with λ-definability [22]. He also published two papers on group theory, including one on finite approximations of continuous groups that was of interest to von Neumann (cf. [24]). Luther P. Eisenhart, who was then head of the mathematics department, urged Turing to stay on for a second year and apply again for the Procter fellowship (worth US$2,000 p.a.). This time, supported by von Neumann who praised his work on almost periodic functions and continuous groups, Turing succeeded in obtaining the fellowship, and so decided to stay the extra year and do a Ph.D. under Church. Proposed as a thesis topic was the idea of ordinal logics that had been broached in Church's course as a way to "escape" Gödel's incompleteness theorems.

Turing, who had just turned 25, returned to England for the summer of 1937, where he devoted himself to three projects: finishing the computability/λ-definability paper, ordinal logics, and the Skewes number. As to the latter, Littlewood had shown that $\pi(x) - \mathrm{li}(x)$ changes sign infinitely often, with an argument by cases, according to whether the Riemann Hypothesis is true or not; prior to that it had been conjectured that $\pi(n) < \mathrm{li}(n)$ for all n, in view of the massive numerical evidence into the billions in support of that.[6] In 1933 Skewes had shown that $\mathrm{li}(n) < \pi(n)$ for some $n < 10_3(34)$ (triple exponential to the base 10) if the Riemann Hypothesis is true. Turing hoped to lower Skewes' bound or eliminate the Riemann Hypothesis; in the end he thought he had succeeded in doing both and prepared a draft but did not publish his work.[7] He was to have a recurring interest in the R.H. in the following years, including devising a method for the practical computation of the zeros of the Riemann zeta function as explained in the article by Andrew R. Booker in this issue of the *Notices*. Turing also made good progress on his thesis topic and devoted himself

[4] *Church's review appeared in* J. Symbolic Logic *2 (1937), 42–43.*

[5] *Gödel accepted the Church-Turing Thesis in that form in a number of lectures and publications thereafter.*

[6] $\mathrm{li}(x)$ *is the (improper) integral from* 0 *to* x *of* $1/\log x$ *and is asymptotic to* $\pi(x)$*, the number of primes* $< x$.

[7] *A paper based on Turing's ideas, with certain corrections, was published after his death by Cohen and Mayhew [4].*

full time to it when he returned to Princeton in the fall, so that he ended up with a draft containing the main results by Christmas of 1937. But then he wrote Philip Hall in March 1938 that the work on his thesis was "proving rather intractable, and I am always rewriting part of it."[8] Later he wrote that "Church made a number of suggestions which resulted in the thesis being expanded to an appalling length." One can well appreciate that Church would not knowingly tolerate imprecise formulations or proofs, let alone errors, and the published version shows that Turing went far to meet such demands while retaining his distinctive voice and original ways of thinking. Following an oral exam in May, on which his performance was noted as "Excellent", the Ph.D. itself was granted in June 1938. Turing made little use of the doctoral title in the following years, since it made no difference for his position at Cambridge. But it could have been useful for the start of an academic career in America. Von Neumann thought sufficiently highly of his mathematical talents to offer Turing a position as his assistant at the Institute. Curiously, at that time von Neumann showed no knowledge or appreciation of his work in logic. It was not until 1939 that he was to recognize the fundamental importance of Turing's work on computability. Then, toward the end of World War II, when von Neumann was engaged in the practical design and development of general purpose electronic digital computers in collaboration with the ENIAC team, he was to incorporate the key idea of Turing's universal computing machine in a direct way.[9]

Von Neumann's offer was quite attractive, but Turing's stay in Princeton had not been a personally happy one, and he decided to return home despite the uncertain prospects aside from his fellowship at King's and in face of the brewing rumors of war. After publishing the thesis work he did no more on that topic and went on to other things. Not long after his return to England, he joined a course at the Government Code and Cypher School, and that was to lead to his top secret work during the war at Bletchley Park on breaking the German Enigma Code. This fascinating part of the story is told in full in Hodges' biography [16], as is his subsequent career working to build actual computers, promote artificial intelligence, theorize about morphogenesis, and continue his work in mathematics. Tragically, this ended with his death in 1954, a probable suicide.

[8] *Hodges [16], p. 144.*

[9] *Its suggested implementation is in the* Draft report on the EDVAC *put out by the ENIAC team and signed by von Neumann; cf. Hodges [16], pp. 302–303; cf. also ibid., p. 145, for von Neumann's appreciation by 1939 of the significance of Turing's work.*

The Thesis: Ordinal Logics[10]

What Turing calls a *logic* is nowadays more usually called a *formal system*, i.e., one prescribed by an effective specification of a language, set of axioms and rules of inference. Where Turing used "L" for logics I shall use "S" for formal systems. Given an effective description of a sequence $\langle S_n \rangle_{n \in N}$ ($N = \{0, 1, 2, \ldots\}$) of formal systems all of which share the same language and rules of inference, one can form a new system $S_\omega = \bigcup S_n$ ($n \in N$), by taking the effective union of their axiom sets. If the sequence of S_n's is obtained by iterating an effective passage from one system to the next, then that iteration can be continued to form $S_{\omega+1}, \ldots$ and so on into the transfinite. This leads to the idea of an effective association of formal systems S_α with ordinals α. Clearly that can be done only for denumerable ordinals, but to deal with limits in an effective way, it turns out that we must work not with ordinals per se, but with *notations for ordinals.* In 1936, Church and Kleene [3] had introduced a system O of constructive ordinal notations, given by certain expressions in the λ-calculus. A variant of this uses numerical codes a for such expressions and associates with each $\alpha \in O$ a countable ordinal $|\alpha|$. For baroque reasons, 1 was taken as the notation for 0, 2^a as a notation for the successor of $|a|$, and $3 \bullet 5^e$ for the limit of the sequence $|a_n|$, when this sequence is strictly increasing and when e is a code of a computable function $\hat{e}$ with $\hat{e}(n) = a_n$ for each $n \in N$. The least ordinal not of the form $|a|$ for some $a \in O$ is the analogue, in terms of effective computability, of the least uncountable ordinal ω_1 and is usually denoted by ω_1^{CK}, where "CK" refers to Church and Kleene. By an *ordinal logic* $S^* = \langle S_a \rangle_{a \in O}$ is meant any means of effectively associating with each $a \in O$ a formal system S_a. Note, for example, that there are many ways of forming a sequence of notations a_n whose limit is ω, given by all the different effectively computable strictly increasing subsequences of N. So at limit ordinals $\alpha < \omega_1^{CK}$ we will have infinitely many representations of α and thus also for its successors. An ordinal logic is said to be *invariant* if whenever $|a| = |b|$ then S_a and S_b prove the same theorems.

In general, given any effective means of passing from a system S to an extension S' of S, one can form an ordinal logic $S^* = \langle S_a \rangle_{a \in O}$ which is such that for each $a \in O$ and $b = 2^a$ the successor of $a, S_b = S'_a$, and is further such that whenever $a = 3 \bullet 5^e$ then S_a is the union of the sequence of $S_{\hat{e}(n)}$ for each $n \in N$. In particular, for systems whose language contains that of Peano Arithmetic PA, one can take S' to be $S \cup \{\mathrm{Con}_S\}$, where Con_S

[10] *The background to the material of this section in Gödel's incompleteness theorems is explained in my piece for the* Notices *[11].*

formalizes the consistency statement for S; the associated ordinal logic S^* thus iterates adjunction of consistency through all the constructive ordinal notations. If one starts with PA as the initial system it may be seen that each S_a is consistent and so S'_a is strictly stronger than S_a by Gödel's second incompleteness theorem. The consistency statements are expressible in $\forall$ ("for all")-form, i.e., $\forall x R(x)$ where R is an effectively decidable predicate. So a natural question to raise is whether S^* is complete for statements of that form, i.e., whether whenever $\forall x R(x)$ is true in N then it is provable in S_a for some $a \in O$. Turing's main result for this ordinal logic was that that is indeed the case, in fact one can always choose such an a with $|a| = \omega + 1$. His ingenious method of proof was, given R, to construct a sequence $\hat{e}(n)$ that denotes n as long as $(\forall x \leq n) R(x)$ holds and that jumps to the successor of $3 \bullet 5^e$ when $(\exists x \leq n) \neg R(x)$.[11] Let $b = 3 \bullet 5^e$ and $a = 2^b$. Now if $\forall x R(x)$ is true, $b \in O$ with $|b| = \omega$. In S_a we can reason as follows: if $\forall x R(x)$ were not true then S_b would be the union of systems that are eventually the same as S_a, so S_b would prove its own consistency and hence, by Gödel's theorem, would be inconsistent. But S_a proves the consistency of S_b, so we must conclude that $\forall x R(x)$ holds after all.

Turing recognized that this completeness proof is disappointing because it shifts the question of whether a $\forall$-statement is true to the question whether a number a actually belongs to O. In fact, the general question, given a, is $a \in O$?, turns out to be of higher logical complexity than any arithmetical statement, i.e., one formed by the unlimited iteration of universal and existential quantifiers, $\forall$ and $\exists$. Another main result of Turing's thesis is that for quite general ordinal logics, S^* can't be both complete for statements in $\forall$-form and invariant. It is for these reasons that above I called his completeness results equivocal. Even so, what Turing really hoped to obtain was completeness for statements in $\forall\exists$ ("for all, there exists")-form. His reason for concentrating on these, which he called "number-theoretical problems", rather than considering arithmetical statements in general, is not clear. This class certainly includes many number-theoretical statements (in the usual sense of the word) of mathematical interest, e.g., those, such as the twin prime conjecture, that say that an effectively decidable set C of natural numbers is infinite. Also, as Turing pointed out, the question whether a given program for one of his machines computes a total function is in $\forall\exists$-form. Of special note is his proof ([23], sec. 3) that the Riemann Hypothesis is a number-theoretical problem in Turing's sense. This was certainly a novel observation for the time; actually, as shown by Georg Kreisel years later, it can even be expressed in $\forall$-form.[12] On the other hand, Turing's class of number-theoretical problems does not include such statements as finiteness of the number of solutions of a diophantine equation ($\exists\forall$) or the statement of Waring's problem ($\forall\exists\forall$).

In section 4 Turing introduced a new idea that was to change the face of the general theory of computation (also known as recursion theory) but the only use he made of it there was curiously inessential. His aim was to produce an arithmetical problem that is not number-theoretical in his sense, i.e., not in $\forall\exists$-form. This is trivial by a diagonalization argument, since there are only countably many effective relations $R(x, y)$ of which we could say that $\forall x \exists y R(x, y)$ holds. Turing's way of dealing with this, instead, is through the new notion of computation relative to an *oracle*. As he puts it:

> Let us suppose that we are supplied with some unspecified means of solving number-theoretical [i.e., $\forall\exists$] problems; a kind of oracle as it were. ... With the help of the oracle we could form a new kind of machine (call them o-machines), having as one of its fundamental processes that of solving a given number-theoretic problem.

He then showed that the problem of determining whether an o-machine terminates on any given input is an arithmetical problem not computable by any o-machine, and hence not solvable by the oracle itself. Turing did nothing further with the idea of o-machines, either in this paper or afterward. In 1944 Emil Post [20] took it as his basic notion for a theory of *degrees of unsolvability*, crediting Turing with the result that for any set of natural numbers there is another of higher degree of unsolvability. This transformed the notion of computability from an absolute notion into a relative one that would lead to entirely new developments and eventually to vastly generalized forms of recursion theory. Some of the basic ideas and results of the theory of effective reducibility of the membership problem for one set of numbers to another inaugurated by Turing and Post are explained in the article by Martin Davis in this issue of the *Notices*.

There are further interesting suggestions and asides in the thesis, such as consideration of possible constructive analogues of the Continuum Hypothesis. Finally (as also mentioned by Barry Cooper in his review article), it contained provocative speculations concerning intuition versus technical

[11] *Note that e is defined in terms of itself; this is made possible by Kleene's index form of the recursion theorem.*

[12] *A relatively perspicuous representation in that form may be found in Davis et al. [6] p. 335.*

ingenuity in mathematical reasoning. The relevance, according to Turing is that:

> When we have an ordinal logic, we are in a position to prove number-theoretic theorems by the intuitive steps of recognizing [natural numbers as notations for ordinals] ...We want it to show quite clearly when a step makes use of intuition and when it is purely formal... It must be beyond all reasonable doubt that the logic leads to correct results whenever the intuitive steps [i.e., recognition of ordinals] are correct.

This Turing had clearly accomplished with his formulation of the notion of ordinal logic and the construction of the particular S^* obtained by iterating consistency statements.

One reason that the reception of Turing's paper may have been so limited is that (no doubt at Church's behest) it was formulated in terms of the λ-calculus, which makes expressions for ordinals and formal systems very hard to understand. He could instead have followed Kleene, who wrote in his retrospective history [17]: "I myself, perhaps unduly influenced by rather chilly receptions from audiences around 1933–35 to disquisitions on λ-definability, chose, after general recursiveness had appeared, to put my work in that format. I cannot complain about my audiences after 1935."

Ordinal Logics Redux

The problems left open in Turing's thesis were attacked in my 1962 paper, "Transfinite recursive progressions of axiomatic theories" [7]. The title contains my rechristening of "ordinal logics" in order to give a more precise sense of the subject matter. In the interests of perspicuity and in order to explain what Turing had accomplished, I also recast all the notions in terms of general recursive functions and recursive notions for ordinals rather than the λ-calculus. Next I showed that Turing's progression based on iteration of consistency statements is not complete for true $\forall\exists$ statements, contrary to his hope. In fact, the same holds for the even stronger progression obtained by iterating adjunction to a system S of the *local reflection principle for* S. This is a scheme that formalizes, for each arithmetical sentence A, that if A is provable in S then A (is true). Then I showed that a progression $S^{(U)}$ based on the iteration of the *uniform reflection principle* is complete for all true arithmetical sentences. The latter principle is a scheme that formalizes, given S and a formula $A(x)$ that if $A(n)$ is provable in S for each n, then $\forall x A(x)$ (is true). One can also find a path P through O along which every true arithmetical sentence is provable in the progression $S^{(U)}$. On the other hand, invariance fails badly in the sense that there are paths P' through O for which there are true sentences in $\forall$-form not provable along that path, as shown in my paper with Spector [12]. The recent book *Inexhaustibility* [13] by Torkel Franzén contains an accessible introduction to [7], and his article [14] gives an interesting explanation (shorn of the off-putting details) of what makes Turing's and my completeness results work.

The problem raised by Turing of recognizing which expressions (or numbers) are actually notations for ordinals is dealt with in part through the concept of *autonomous progressions of theories*, obtained by imposing a boot-strap procedure. That allows one to go to a system S_a *only* if one already has a proof in a previously accepted system S_b that $a \in O$ (or that a recursive ordering of order type corresponding to a is a well-ordering). Such progressions are not complete but have been used to propose characterizations of certain informal concepts of proof, such as that of finitist proof (Kreisel [18], [19]) and predicative proof (Feferman [8], [9]).

References

[1] A. Church, An unsolvable problem of elementary number theory, *Amer. J. of Math.* **58** (1936), 345–63. Reprinted in Davis [5].

[2] ——, A note on the Entscheidungsproblem, *J. Symbolic Logic* **1** (1936), 40–41; correction, ibid., 101–2. Reprinted in Davis [5].

[3] A. Church and S. C. Kleene, Formal definitions in the theory of ordinal numbers, *Fundamenta Mathematicae* **28** (1936), 11–21.

[4] A. M. Cohen and M. J. E. Mayhew, On the difference $\pi(x) - \mathrm{li}(x)$, *Proc. London Math. Soc.* **18**(3) (1968), 691–713; reprinted in Turing [24].

[5] M. Davis, *The Undecidable. Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions*, Raven Press, Hewlett, NY, (1965).

[6] M. Davis, Yu. Matijasevič, and J. Robinson, Hilbert's tenth problem. Diophantine equations: positive aspects of a negative solution, *Mathematical Developments Arising From Hilbert Problems*, (F. Browder, ed.), Amer. Math. Soc., Providence, RI, (1976), 323–78.

[7] S. Feferman, Transfinite recursive progressions of axiomatic theories, *J. Symbolic Logic* **27** (1962), 259–316.

[8] ——, Systems of predicative analysis, *J. Symbolic Logic* **29** (1964), 1–30.

[9] ——, Autonomous transfinite progressions and the extent of predicative mathematics, *Logic, Methodology and Philosophy of Science* III, (B. van Rootselaar and J. F. Staal, eds.), North-Holland, Amsterdam (1968), 121–35.

[10] ——, Turing in the land of $O(z)$, in Herken [15], 113–47.

[11] ——, The impact of the incompleteness theorems on mathematics, *Notices Amer. Math. Soc.* **53** (April 2006), 434–9.

[12] S. Feferman and C. Spector, Incompleteness along paths in recursive progressions of theories, *J. Symbolic Logic* **27** (1962), 383–90.

[13] T. FRANZÉN, *Inexhaustibility. A non-exhaustive treatment*, Lecture Notes in Logic **28** (2004), Assoc. for Symbolic Logic, A. K. Peters, Ltd., Wellesley (distribs.).

[14] ____, Transfinite progressions: A second look at completeness, *Bull. Symbolic Logic* **10** (2004), 367–89.

[15] R. HERKEN (ed.), *The Universal Turing Machine. A Half-Century Survey*, Oxford University Press, Oxford (1988).

[16] A. HODGES, *Alan Turing: The Enigma*, Simon and Schuster, New York, 1983. New edition, Vintage, London, 1992.

[17] S. C. KLEENE, Origins of recursive function theory, *Ann. History of Computing* **3** (1981), 52–67.

[18] G. KREISEL, Ordinal logics and the characterization of informal concepts of proof, *Proc. International Congress of Mathematicians at Edinburgh* (1958), 289–99.

[19] ____, Principles of proof and ordinals implicit in given concepts, *Intuitionism and Proof Theory*, (J. Myhill et al., eds.), North-Holland, Amsterdam, (1970) 489–516.

[20] E. POST, Recursively enumerable sets and their decision problems, *Bull. Amer. Math. Soc.* **50** (1944), 284–316.

[21] A. M. TURING, On computable numbers, with an application to the Entscheidungsproblem, *Proc. London Math. Soc.* **42**(2) (1936–37), 230–65; correction, ibid. **43** (1937), 544–6. Reprinted in Davis [5] and Turing [25].

[22] ____, Computability and λ-definability, *J. Symbolic Logic* **2** (1937), 153–63. Reprinted in Davis [5] and Turing [25].

[23] ____, Systems of logic based on ordinals, *Proc. London Math. Soc.* (2) (1939), 161–228. Reprinted in Davis [5] and Turing [25].

[24] ____, *Pure Mathematics* (J. L. Britton, ed.), *Collected Works of A. M. Turing*, Elsevier Science Publishers, Amsterdam, (1992).

[25] ____, *Mathematical Logic* (R. O. Gandy and C. E. M. Yates, eds.), *Collected Works of A. M. Turing*, Elsevier Science Publishers, Amsterdam, (2001).

Turing and the Riemann Hypothesis

Andrew R. Booker

Alan Turing's final research paper[1] [11] described a numerical method for verifying the Riemann hypothesis and its implementation on the Manchester Mark I, one of the earliest general purpose digital computers. Turing writes in his introduction

> The calculations had been planned some time in advance, but had in fact to be carried out in great haste. If it had not been for the fact that the computer remained in serviceable condition for an unusually long period from 3 p.m. one afternoon to 8 a.m. the following morning it is probable that the calculations would never have been done at all. As it was, the interval $2\pi.63^2 < t < 2\pi.64^2$ was investigated during that period, and very little more was accomplished.

The modesty of this last sentence notwithstanding, Turing's paper is an important contribution to number theory that continues to have relevance today; indeed, we are fortunate that the Manchester computer remained serviceable for so long on that day, for otherwise Turing may never have published his results! The goal of this article is to describe the method and some recent developments in a historical context.

Background

We begin with a very brief introduction to the Riemann hypothesis and some associated computational aspects; for a full account, including its importance in number theory and recent attempts at proof, see the excellent survey article by Conrey [4].

The ζ-function is defined for complex numbers s with real part $\Re(s) > 1$ by the series

$$\zeta(s) := \sum_{n=1}^{\infty} \frac{1}{n^s}, \tag{1}$$

which converges absolutely. As discovered by Riemann, it has analytic continuation to $\mathbb{C}$, except for a simple pole at $s = 1$. Moreover, a functional equation relates the values at s and $1-s$: If $\gamma(s) := \pi^{-s/2}\Gamma(s/2)$ and $\Lambda(s) := \gamma(s)\zeta(s)$ then

$$\Lambda(s) = \Lambda(1-s). \tag{2}$$

The Riemann hypothesis is the conjecture that all zeros of the modified function $\Lambda(s)$ have real part exactly $\frac{1}{2}$. All that is known at present, however, is that the real parts lie in the open interval $(0, 1)$.

Since $\Lambda(s)$ is real for s on the real axis, the zeros come in complex-conjugate pairs $s, \bar{s}$, so it suffices to consider here only the ones in the upper half plane. The number of zeros with imaginary part $\Im(s) \in (0, t]$, denoted by $N(t)$, is roughly $\theta(t)/\pi + 1$, where $\theta(t)$ is the phase of $\gamma(\frac{1}{2} + it)$, i.e., the continuous function such that $\theta(0) = 0$ and

$$\gamma(\tfrac{1}{2} + it) = |\gamma(\tfrac{1}{2} + it)|e^{i\theta(t)}. \tag{3}$$

Andrew R. Booker is a lecturer at the University of Bristol. His email address is `andrew.booker@bristol.ac.uk`.

[1] *A popular account of some of his ideas on computability appeared the following year in [12].*

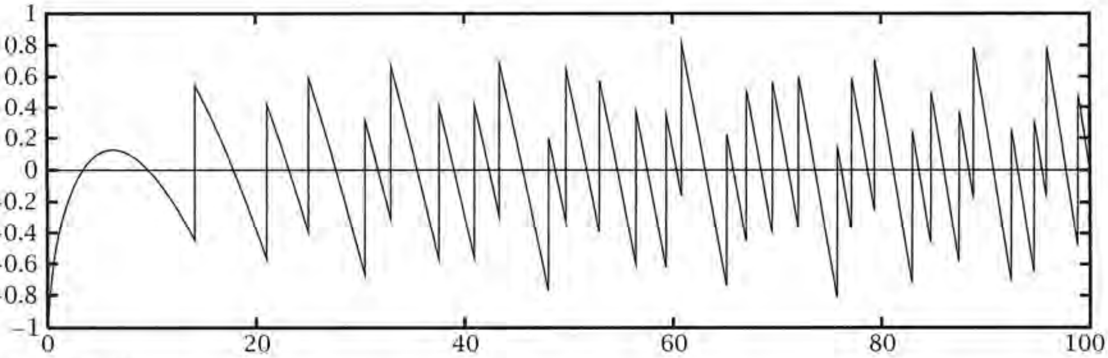


Figure 1. $S(t)$.

This may be computed quickly for large $t > 0$ by the asymptotic formula

$$\frac{\theta(t)}{\pi} + 1 \approx \frac{t}{2\pi}\log\frac{t}{2\pi e} + \frac{7}{8}. \tag{4}$$

In particular, $\Lambda(s)$ has many zeros.

The difference

$$S(t) := N(t) - \Big(\frac{\theta(t)}{\pi} + 1\Big) \tag{5}$$

between $N(t)$ and the expected count is a function that seems to vary unpredictably, as can be seen in Figure 1. This strange[2] behavior, and our incomplete understanding of it, lies at the heart of what makes the Riemann hypothesis a difficult problem and numerical computation a useful tool.

An important ingredient when doing numerics is an algorithm for computing the Λ-function at arguments $s = \frac{1}{2} + it$. However, since $|\gamma(\frac{1}{2} + it)|$ decreases exponentially for large t, one usually works instead with the function $Z(t) := e^{i\theta(t)}\zeta(\frac{1}{2} + it)$, which is real-valued for $t \in \mathbb{R}$ and has the same zeros as $\Lambda(\frac{1}{2} + it)$. A formula for $Z(t)$, known to Riemann and rediscovered by Siegel, is the following.

$$Z(t) \approx 2 \sum_{n=1}^{\lfloor\sqrt{t/2\pi}\rfloor} n^{-1/2}\cos(\theta(t) - t\log n). \tag{6}$$

The error of the approximation is no worse than $O(t^{-1/4})$, so that (6) becomes more accurate for larger t; moreover, there is an asymptotic expansion for the error term, giving better accuracy yet. For small values of t, the error in the Riemann-Siegel formula is too large, and one usually prefers a different technique, known as Euler-Maclaurin summation, which allows for high accuracy at the expense of a longer running time.

Turing's Interest in the Riemann Hypothesis

According to Hodges' definitive biography [6], Turing became interested in the Riemann hypothesis while still a student. Curiously, he seems to have believed it to be false; indeed, it is clear from [11] that he had hoped to find a counterexample. In 1939, back in Cambridge, he conceived of an analog machine to aid with the calculations necessary for numerically checking the hypothesis. The design of the machine, whose blueprint is reproduced on the cover and pages 1186–1187, called for an assembly of eighty gears of precise ratios and a counterweight, which would physically perform the sum in (6). Turing won a grant for £40 from the Royal Society to cover the cost of its construction and got as far as manually cutting some of the gears, which would often end up on the floor of his room. However, World War II intervened before the work was completed, and Turing would have other important contributions to make.

By the time that he returned to the problem, in June 1950, digital computers had advanced to the point that it was practical, if only barely so, to consider much more than was possible with any analog machine—testing the Riemann hypothesis algorithmically, with no human intervention. Indeed, this is an important aspect of Turing's method which should not be overlooked.[3] Although Turing's numerical results were modest—Titchmarsh had by 1936 achieved nearly the same range by more conventional means—it wasn't long before Lehmer extended his calculations to ranges well out of reach of hand computation. However, that this would be the case may have been far from obvious in 1950; few at the time could have anticipated the economies of scale in speed, reliability, and availability of computing technology that would be achieved, forever rendering human computers obsolete.[4] As it was, the practical issues that Turing faced, described in detail in [11], were formidable compared to today's technology.

[2] *One might substitute the word* random *here: Selberg showed that the values of $S(t)/\sqrt{\log\log t}$, as $t \to \infty$, are normally distributed.*

[3] *It was also part of the larger consideration of the extent to which machines could think and act autonomously, a question that captured Turing's keen interest.*

[4] *Up to the 1940s, the word* computer *referred to a human who performed computations with the aid of a calculating device. With the advent of electronic machines and stored programs, the job of the human shifted to that of* programmer. *Turing employs both the original and modern usages in [11].*

Hodges speculates that Turing rushed [11] to publication, worried that he would be sent to prison. What is clear is that he was dissatisfied with the results. Unfortunately, we will never know the true extent of his intentions.

The Method

Turing's approach follows earlier computations by Gram, Backlund, Hutchinson, and Titchmarsh. First, one locates many zeros on the line $\Re(s) = \frac{1}{2}$ up to a given height T by computing $Z(t)$ and noting its changes of sign. Second, one shows that the computed list of zeros is complete (meaning that the Riemann hypothesis is true up to height T), by determining $N(T)$ via an auxiliary computation.

Turing made contributions to both aspects. In [10] he introduced an algorithm for computing the Z-function that was intended to be used in the intermediate range, between those of the Riemann-Siegel formula and Euler-Maclaurin summation. However, with better error terms known today and improvements in computing technology, that gap has been closed otherwise. On the other hand, his technique for determining $N(T)$ was of more lasting value and is what is usually meant when referring simply to "Turing's method".

The authors prior to Turing used an ad hoc approach, described in detail by Edwards [5, §6.7]; it was both computationally expensive and not guaranteed to work for any given T. Turing's method relies instead on the fact, first due to Littlewood, that the average value of $S(t)$, for t ranging over the interval $[0, T]$, tends to 0 as T grows. Thus, the graph of $S(t)$ tends to oscillate around 0, as is visible in Figure 1. Now, if one imagines plotting Figure 1 using equation (5) and the *measured* data for $N(t)$, any zeros that had been missed would skew the graph, i.e., it would begin to oscillate around an integer less than 0, corresponding to the number of missing zeros. (Note that when locating zeros by sign changes, one always misses an even number of them.)

To make this precise, one needs an explicit form of Littlewood's theorem. This is one of the main results of [11], where Turing proved the estimate

$$(7) \qquad \left| \int_T^{T+h} S(t)\,dt \right| \le 2.3 + 0.128 \log \frac{T+h}{2\pi},$$

valid for all $h > 0$ and $T > 168\pi$. With (7) in hand, one entertains the hypothesis that at least one zero up to height T has been overlooked and computes the integral using the numerically measured values of $N(t)$, with the extra zero thrown in. If it turns out that there really is no missing zero, then (7) will be contradicted with a value of h on the order of $c \log T$ for a small number c. Thus, roughly speaking, in order to certify complete the list of zeros up to T, one needs knowledge of the ζ-function up to height about $T + c \log T$. When T is large, that is a negligible price to pay compared to the total computation.

Turing's proof of (7) is elegant and remains essentially unchanged in all subsequent generalizations. The bound is not sharp,[5] however, and the constants were later improved by Lehman, who also corrected a few errors in the details. Nevertheless, (7) is more than sufficient for numerics; in fact, in modern verifications of the Riemann hypothesis, Turing's method is considered an automatic check, and one can concentrate on the business of locating the zeros as quickly as possible.

Recent Developments

The more than half century following Turing's death has seen many developments in computational aspects of the Riemann hypothesis and related problems. In fact, Turing's method is arguably the first in a long line of papers in the area of computational analytic number theory; see [8] for a recent survey.

Concerning the Riemann hypothesis, an essentially optimal algorithm (in terms of speed) for computing the ζ-function was developed by Odlyzko and Schönhage [7]. It uses the Fast Fourier Transform and computes many values of $Z(t)$ in *average* time $O(t^\varepsilon)$ per value, compared to the roughly $\sqrt{t}$ steps needed for a single evaluation using the Riemann-Siegel formula. The algorithm has led to computations of the ζ-function on a much larger scale than Turing could have envisioned; in particular, the Riemann hypothesis has now been verified up to the ten trillionth zero! Turing's method remains a small but essential ingredient in those investigations.

Perhaps more importantly, the same computations have aided in the discovery of links between the ζ-function and random matrix theory, which has in turn led to a flurry of recent work. A strong argument can be made that the eventual proof of the Riemann hypothesis will require a deeper understanding of this connection. See [4] for a description of these exciting developments.

In the same vein, number theorists today recognize that the ζ-function is just one of a large class of important generating functions, known as L-functions. Many of the conjectures for ζ, including the Riemann hypothesis and connections with

[5] *The coefficient of* $\log \frac{T+h}{2\pi}$ *in Turing's estimate is closely related to knowledge about the growth rate of* $Z(t)$ *as* $t \to \infty$. *The Lindelöf hypothesis, which is the conjecture* $Z(t) = O(t^\varepsilon)$, *is equivalent to the integral being* $o(\log(T+h))$ *as* $T + h \to \infty$. *The Riemann hypothesis, which in turn implies the Lindelöf hypothesis, yields the stronger bound* $O\left(\frac{\log(T+h)}{\log\log(T+h)^2}\right)$. *Heuristic arguments based on random matrix theory suggest that the true maximum size of the integral is closer to* $\sqrt{\log(T+h)}$.

random matrix theory, are expected to hold true for these functions as well. Very recently, Turing's method has been extended to all L-functions in [2]. It is interesting to note that few of the known techniques in analytic number theory apply in such wide generality; the fact that Turing's method does demonstrates how fundamental it is.

Finally, the novelty of Turing's method is further underscored by the fact that it was rediscovered some forty years later in the seemingly unrelated context of computing the spectrum of the Laplace operator on hyperbolic manifolds. This has had several applications in number theory and high energy physics; see [9] for a nice survey and [1] for an interesting application to cosmology. Unfortunately, the papers in the subject generally use the method without proper attribution to Turing. It would be good to have the record set straight. To that end, a rigorous treatment of the simplest example, much in the style of Turing, will appear in [3].

References

[1] RALF AURICH, SVEN LUSTIG, FRANK STEINER, and HOLGER THEN, Hyperbolic universes with a horned topology and the cosmic microwave background anisotropy, *Classical Quantum Gravity* **21** (2004), no. 21, 4901–25.

[2] ANDREW R. BOOKER, Artin's conjecture, Turing's method, and the Riemann hypothesis, *Exp. Math.*, (to appear, 2006).

[3] ANDREW R. BOOKER and ANDREAS STRÖMBERGSSON, *Effective computation of Maass cusp forms*, II, in preparation.

[4] J. BRIAN CONREY, The Riemann hypothesis, *Notices Amer. Math. Soc.* **50** (2003), no. 3, 341–53.

[5] H. M. EDWARDS, *Riemann's Zeta Function*, Dover Publications Inc., Mineola, NY, 2001. Reprint of the 1974 original [Academic Press, New York; MR0466039 (57 #5922)].

[6] A. HODGES, *Alan Turing: The Enigma*, Simon & Schuster, New York, 1983. New edition, Vintage, London, 1992.

[7] A. M. ODLYZKO and A. SCHÖNHAGE, Fast algorithms for multiple evaluations of the Riemann zeta function, *Trans. Amer. Math. Soc.* **309** (1988), no. 2, 797–809.

[8] MICHAEL RUBINSTEIN, Computational methods and experiments in analytic number theory, *Recent Perspectives in Random Matrix Theory and Number Theory*, London Math. Soc. Lecture Note Ser., vol. 322, Cambridge Univ. Press, Cambridge, 2005, pp. 425–506.

[9] PETER SARNAK, Spectra of hyperbolic surfaces, *Bull. Amer. Math. Soc.* (N.S.) **40** (2003), no. 4, 441–78 (electronic).

[10] A. M. TURING, A method for the calculation of the zeta-function, *Proc. London Math. Soc.* **48**(2) (1943), 180–97.

[11] ____, Some calculations of the Riemann zeta-function, *Proc. London Math. Soc.* **3**(3) (1953), 99–117.

[12] ____, Solvable and unsolvable problems, *Science News* **31** (1954), 7–23.

The Man Who Knew Too Much: Alan Turing and the Invention of the Computer

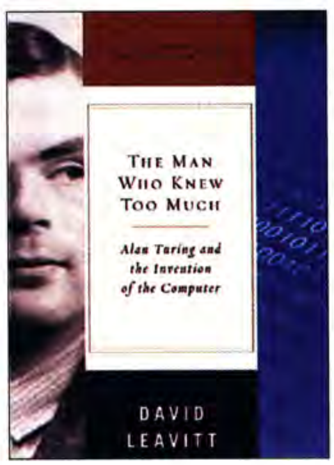
Reviewed by S. Barry Cooper

The Man Who Knew Too Much: Alan Turing and the Invention of the Computer
David Leavitt
Great Discoveries Series, W. W. Norton
288 pages, US$22.95
December 2005, ISBN 0-393-05236-2

Mathematicians like to think of themselves as seekers after truth. At the same time, there is an optimistic modeling of mathematics as a rational activity based upon generally shared foundations.

Alan Turing—along with Kurt Gödel and Alonzo Church and others—was one of those metamathematical pioneers working in the 1930s who showed that the actual picture was in some ways more mundane, and in others very much stranger. And what makes Turing's work so interesting to people outside mathematics is the extent to which his mathematical investigations were tied up with his own personality and tortuous personal affairs. In many ways, Turing's mathematics was not just about what mathematicians and computers can and cannot do, but seems to many to be highly relevant to his own life and all too early death.

David Leavitt's highly readable account of Turing's life and work is yet one more example of this attention from nonmathematicians, and one that will be greeted with the usual mixed feelings by those of us who work professionally in the field. We are happy to see mathematics and mathematicians given the popular science treatment—even



when, as here, it is marred by the occasional technical gaffe. But there are many points at which David Leavitt, who (the publisher tells us) "teaches creative writing at the University of Florida," tests the reader's patience with his extra-mathematical takes on Turing's career, particularly in regard to the relationship between Turing's sexuality and his scientific work. He certainly seems to have tested that of Andrew Hodges, the perceptive and sensitive author of the definitive Turing biography (one which must be a front-runner for the best biography of a mathematician ever written). In his *Scientific American* review (January 22, 2006) of *The Man Who Knew Too Much* Hodges describes, as someone who has moved beyond 1960s posturing, how "Leavitt's focus ...is on Turing as the gay outsider, driven to his death" with no "opportunity ...lost to highlight this subtext"; how it is "a survey of a field long cultivated by other hands, devoid of new witnesses" (most of Leavitt's factual account of Turing is based on already published sources); but—and how could one not enjoy this readable page-turner of a book?—he concludes, the book "is one that many will find congenial and that will at least introduce new readers to the still tingling enigma of Alan Turing."

Barry Cooper is professor of mathematical logic at the University of Leeds, UK. His may be contacted via s.b.cooper@leeds.ac.uk *or* http://www.amsta.leeds.ac.uk/~pmt6sbc.

There will, of course, be more radical and variously dissenting views on what might seem the much-hyped status of Alan Turing, in particular from those who point to the paucity of his published mathematical works, and these technically surpassed by a legion of worthy researchers for whom we will never see popular biographies. For the dissenters the continuing fascination exercised by Turing and his work is a byproduct of the mythology generated by his Bletchley Park codebreaking (shortening the Second World War by two years, it is estimated); of his controversial role as "father of the computer"; of his persecution as an openly gay man in post-war Britain; and of his mysterious premature death from the eating of an apple dipped in cyanide (which Leavitt, building on what Hodges tells us, bizarrely links to Turing's captivation by the Disney version of *Snow White and the Seven Dwarfs*).

But then, the work of Turing—and, for that matter, that of Gödel, who had great respect for Turing—is a far cry from the contemporary model involving slabs of technically proficient mathematics crafted by committees of collaborating mathematicians. No doubt, startlingly original discoveries do emerge from within quite different research paradigms, but Turing's vision was very much linked to his own peculiarly individual solitariness. If one opens the recent volume edited by Christof Teuscher, *Alan Turing: Life and Legacy of a Great Thinker* (Springer, 2004), one discovers a whole range of basic everyday issues, still scientifically relevant, to which Turing made clarifying and seminal contributions. Artificial intelligence? His most visible legacy is the Turing Test. Quantum theory related to mental functionality? Turing was there at the beginning of the discussion. The theoretical limits of machine intelligence? Turing's 1939 paper is full of key ideas, often giving rise to whole new areas of research. Some of these he never returned to (like his influential invention of the oracle model for interactive computation), while others preoccupied him until the end (such as computers that, like humans, make mistakes). And more. Measuring the complexity of computations? Turing provided a basic computational model upon which this could be based. Emergence in nature? Here we have Turing's ground-breaking 1952 paper on "The chemical basis of morphogenesis". As Hodges says in his *Scientific American* review, "Turing's reputation is now solidly underpinned by the vindication of his vision."

However, for most of us, preoccupied with the increasing pressures to conform to algorithmic performance indicators (devised by people for whom Turing's work might be salutary, but in reality has passed them by) it is Thomas Kuhn's "normal science" that dominates our professional lives. In the short term, "vision" does not cut much ice with promotion or appointments committees. Time spent clarifying deep basic questions and formulating new concepts and corresponding technical frameworks is less surely rewarded than work within well-established scientific frameworks, replete with "open problems" based on familiar parameters and established technical repertoires. The appeal of fashionable new areas, such as data mining, algorithms for genetic research, and so forth, may be viewed by the powers-that-be as much more exciting than number theory. But amongst mathematicians, it is commonly held that analysis or number theory are "deeper" than computer science or other newly emergent areas. This is a deepness that can even be marketed (like the proof of Fermat's Last Theorem, or, more lucratively, John Nash's work), but not usually for the light it throws on the world around us. More often, it is mathematics as extreme sport that catches the popular imagination. Of course, esoteric and highly abstract "normal science" does have a habit of throwing up unexpected and fundamentally important applications, but that is hard to explain to the nonspecialist.

Turing's deepness (as Hodges and Leavitt remind us) was based on an almost tactile, but at the same time quite abstract, relationship with the world he lived in. For Leavitt, Alan Turing psychologically *identifies* with the computing machines he studied in such theoretical and practical detail. This was first apparent in his 1937 paper, giving a negative solution to Hilbert's *Entscheidungsproblem*, where he bases his Turing machine model upon a detailed analysis of how a computing clerk, complete with states of mind, might perform. In Chapter 6 ("The Electronic Athlete"), I found Leavitt surprisingly convincing in arguing for Turing's identification with computing machines in a complex world, in which his own homosexuality contributed so much to his personal complexity of context. A Turing machine may compute with surprising sophistication: An *Alan* Turing machine needed to interact with the perplexities of an incongruous real world, and (as Turing himself believed) had to be enabled to make mistakes in order to be intelligent. Leavitt (p. 269) quotes Turing's letter to his friend Norman Routledge, in which he tells him of his impending Manchester court case for "gross indecency" with another male:

> I shall shortly be pleading guilty to a charge of sexual offences with a young man. The story of how it all came to be found out is a long and fascinating one, which I shall have to make into a short story one day, but have not time to tell you now. No doubt I shall emerge from it all a different man, but quite who I've not found out. …I'm rather afraid that

the following syllogism may be used by some in the future:

Turing believes machines think

Turing lies with men

Therefore machines do not think

Leavitt notes: "It is signed, 'Yours in distress, Alan'".

As Leavitt points out, Turing's 1937 paper is phrased in terms of the person within the machine, in apparent contrast to the present day more extensional focus on the mechanical content of the complex. But as Turing's student Robin Gandy points out in his 1988 article "The confluence of ideas in 1936", Turing's approach is a potent one. Typically, Turing starts with a more basic question than that asked by other authors. Not "What is a computable function?" But (Gandy, p. 249):

> The real question at issue is 'What are the possible processes which can be carried out in computing a [real] number?'

The result was a new model of computability very different from that previously thrown up by the logicians. The new model was instrumental in convincing the previously skeptical Gödel that the notion of in-principle computability had indeed been captured and has proved more useful than any other in the subsequent development of theoretical computer science. In following through the parallel between human complexity and that of the wider universe, the Turing model has played a key role in attempts to understand both via the mediating role of the machine.

Gödel himself, it seems, obtained his Incompleteness Theorem via an initial attempt to validate Hilbert's vision of a mathematics tamed within formal systems. One is struck by the fact that both Turing and Gödel set out to try to expand the boundaries of the mechanical within the (mathematical) world, and in so doing obtained such dramatic intimations of the nonmechanical nature of much of the universe around us. Through them, the dichotomy between the mechanical and the complex, between determinism and randomness, or between chaos and emergence, have taken more specific forms as that between the local and the global, and, more specifically, between the computable and our growing understanding of different levels of incomputability.

In his 1939 paper, based on his Ph.D. thesis written under the supervision of Alonzo Church during his stay at Princeton, Turing takes things much further. This can be seen as part of an attempt, which would occupy him for much of his remaining fifteen years of life, to extend the bounds of effectiveness beyond those he himself had established. In this marvelous paper he once again pursues his constructive agenda with unexpected outcomes. Ever alive to the real-world context of his work, this is what Turing (pages 134–5), says about the underlying meaning of his paper:

> Mathematical reasoning may be regarded ...as the exercise of a combination of ...*intuition* and *ingenuity*. ...In pre-Gödel times it was thought by some that all the intuitive judgments of mathematics could be replaced by a finite number of ...rules. The necessity for intuition would then be entirely eliminated. In our discussions, however, we have gone to the opposite extreme and eliminated not intuition but ingenuity, and this in spite of the fact that our aim has been in much the same direction.

So he is addressing the familiar mystery of how we often arrive at a mathematical result via what seems like a very unmechanical process, but then promptly retrieve from this a proof that is quite standard and communicable to other mathematicians. Another celebrated mathematician, well-known for his interest in the role of intuition in the mathematician's thinking, was Poincaré. A few years after Turing wrote the above passage, Jacques Hadamard in *The Psychology of Invention in the Mathematical Field* recounts how Poincaré got stuck on a problem (concerning elliptic functions):

> At first Poincaré attacked [a problem] vainly for a fortnight, attempting to prove there could not be any such function ...[quoting Poincaré:]
>
> 'Having reached Coutances, we entered an omnibus to go some place or other. At the moment when I put my foot on the step, the idea came to me, without anything in my former thoughts seeming to have paved the way for it ...I did not verify the idea ...I went on with a conversation already commenced, but I felt a perfect certainty. On my return to Caen, for conscience' sake, I verified the result at my leisure.'

The experience of Poincaré may have been a dramatic one, but not one unfamiliar to the working mathematician. Who else but Turing would have attempted a mathematical explanation at that time? His argument is still not widely known (being beyond what mathematical content one can ask from Leavitt's book). Its significance is certainly not understood, except by those at ease with both the mathematics and with thinking about the world in the sort of basic terms that came naturally to Turing. What is significant is Turing's identification of incomputability with aspects of what happens in the human mind, so anticipating more recent—and more controversial—thinking on this topic. Of course, in the end Turing was faced with the jungle of incomputability in his own personal life. And when Leavitt points to the role of Turing's homosexuality in counter-posing instinct (in the form of "mechanical" basic drives) with emergent and conflicting conventional values, the relevance to his scientific life cannot be easily dismissed. Particularly interesting is Leavitt's account of Alan Turing's participation in Wittgenstein's Foundations of Mathematics course upon his return to Cambridge from Princeton towards the end of 1939. This is one of the rare occasions when we get something more than Andrew Hodges offers. Leavitt describes both Turing and Wittgenstein as "pragmatists", but with Wittgenstein taking the openly radical position that reasoning is not algorithmic and going so far as to say that paradoxes are not threatening because outside the logical formalism we have thought processes that are more powerful than the blind formal processes. But Turing's radicalism is a reluctant one, always coming from working within the machine—which may be why Turing's contribution has more fundamentally changed how we view the world. Turing is described as persistently algorithmic to the extent that he takes paradoxes arising from logic more seriously, and tries to rescue the algorithm. Both Wittgenstein and Turing are characterized as being rooted in the real world—unlike the typical professional logician or computability theorist of today—but as differing on how one balances algorithmic content and higher physical and mental processes. Here is how Leavitt sets the scene for his comments on their contrasting world-views:

> One of Wittgenstein's ambitions was to compel his students to recognize the importance of common sense even in philosophical enquiry. ('Don't treat your common sense like an umbrella,' he told them. 'When you come into a room to philosophize, don't leave it outside but bring it in with you.') Nor was it accidental that of all the participants in the seminar, it was Turing he singled out, time and again, to serve as the representative of what might be called the logicist position; Wittgenstein was, in his own words, always trying to 'tempt' Turing towards making claims that favored logic over common sense (though not always with success). As a practicing mathematician, Turing could be counted on to reiterate the traditional postulates of his discipline and in so doing give Wittgenstein the opportunity to pull the rug out from under them. Church, or someone like him, would have made a more convenient whipping boy, and had Wittgenstein known more about the unorthodoxy of some of Turing's ideas, he might have taken a different tack.

This is popular science writing of a quite high order. In *The Man Who Knew Too Much* there is certainly enough on target to make it a thoroughly recommendable book for the student or busy professional without the attention span or time to take on Andrew Hodges' more demanding 600 pages. In many ways, Turing's inner contradictions mirrored those of our own age. On the one hand, Solomon Feferman, writing in 1988 (*Turing in the Land of O(z)*, pages 131–2) confirms a generally held view:

> Turing, as is well known, had a mechanistic conception of mind, and that conviction led him to have faith in the possibility of machines exhibiting intelligent behavior.

On the other, we have Alan Turing's interest in quantum theory, found in his schoolboy writings, and re-emerging in his late postcards to Robin Gandy. In between there came his 1944-48 experiences of the ACE (Automatic Computing Engine) project, and his interest in such possibilities as machines that make mistakes. In a talk to the London Mathematical Society, February 20, 1947, he admits "... if a machine is expected to be infallible, it cannot also be intelligent. There are several theorems which say almost exactly that." And Turing also anticipated the importance now given to connectionist models of computation (as described in a 1998 article "On Alan Turing's anticipation of connectionism" by Jack Copeland and Diane Proudfoot).

A "mechanistic conception of mind" maybe, but no crude extension of the Church-Turing thesis in sight, even at a time when Turing had a huge personal investment in the development of computing machinery. Turing never ceased to emphasize the importance of computational context (quoting the LMS talk again):

> No man adds very much to the body of knowledge. Why should we expect more of a machine? Putting the same point differently, the machine must be allowed to have contact with human beings in order that it may adapt itself to their standards.

The mysteries Turing grappled with remain. To what extent is the logic of a Turing machine sufficient to capture the workings of a human brain? What is the nature of the mechanical in the physical world? And what relationship does this have to the mind?

We still try to set aside part of our professional days for the search for truth. But, partly through Turing, Gödel, and their contemporaries, we recognize that *science* is concerned with the extraction of algorithmic content and that our grasp on truth beyond that is hard won. Mathematical perceptions without proofs remain a fairly personal property, and scientific theories that, like psychoanalysis, do not make predictions do not really qualify as science. But we also know that algorithmic content often emerges in a very nonalgorithmic way, as did the proof Poincaré found as he entered the bus at Coutances.

Today, computability theoretic puzzles still lie at the core of many scientific controversies. Computability (or recursion theory as it is still sometimes termed) has come a long way since its origins in mathematical logic. Like a cuckoo in the logical nest, it made itself at home there but could never be completely constrained within logical preoccupations with language and human reasoning. This book, and the widespread interest in the life and work of Alan Turing, could help bring this fundamentally fascinating aspect of present-day science to a wider readership.

W H A T   I S . . .
Turing Reducibility?

Martin Davis

Intuitively, to say that something is *computable* means that there is an algorithm for computing it. Computability theory makes this concept precise. So, in particular, we are enabled to define with full rigor what it means to say that a function defined on the natural numbers and with natural number values is *computable*. Likewise a set of natural numbers is *computable* if its characteristic function (defined to be 1 for members and 0 for non-members) is computable. Having such definitions makes it possible to prove that certain objects are not computable. A fundamental result is that there is a computable function whose range (the set of all values that it assumes) is not computable. Sets that are the range of a computable function are called *recursively enumerable*. (The empty set is also considered to be recursively enumerable.) The fact that there is a recursively enumerable set that is not computable is a special case of a more general result that will be explained later in this article. It is the use of this fact that has made it possible to prove that a number of important mathematical problems are unsolvable, that algorithms that mathematicians had been seeking simply do not exist. Among these problems are Hilbert's 10th Problem (to decide whether a given Diophantine equation has solutions), the word problem for groups (to decide whether a given product of generators and their inverses is the identity element of a group defined by a finite set of equations between such products), and the homeomorphy problem (to decide whether the topological spaces defined by a given pair of simplicial complexes are homeomorphic).

Martin Davis is professor emeritus of mathematics and computer science, New York University, and is a visiting scholar in mathematics at the University of California, Berkeley. His email address is `martin@eipye.com`.

The concept of *Turing reducibility* has to do with the question: can one non-computable set be more non-computable than another? In a rather incidental aside to the main topic of Alan Turing's doctoral dissertation (the subject of Soloman Feferman's article in this issue of the *Notices*), he introduced the idea of a computation with respect to an oracle. An *oracle* for a particular set of natural numbers may be visualized as a "black box" that will correctly answer questions about whether specific numbers belong to that set. We can then imagine an oracle algorithm whose operations can be interrupted to query such an oracle with its further progress dependent on the reply obtained. Then for sets A, B of natural numbers, A is said to be *Turing reducible* to B if there is an oracle algorithm for testing membership in A having full recourse to an oracle for B. The notation used is: $A \leq_t B$. Of course, if B is itself a computable set, then nothing new happens; in such a case $A \leq_t B$ just means that A is computable. But if B is non-computable, then interesting things happen.

As the notation suggests, Turing reducibility is a partial order. If sets A, B are each Turing reducible to the other, they are said to be Turing equivalent, written $A \equiv_t B$. And if A is Turing reducible to B but not conversely we write $A <_t B$. By considering all oracle algorithms having access to an oracle for a particular set A, one can construct a new set A' that contains all the information concerning membership in any set Turing reducible to A. (The construction is analogous to that of a "universal" Turing machine.) The operation $'$ is called the *jump* because it can easily be proved (by using a Cantor-style diagonal argument) that $A <_t A'$. Setting $A =$

$\varnothing$, the set $\varnothing'$ provides an example of a recursively enumerable set that is not computable. Iterating the jump operation, one obtains the sequence of more and more unsolvable problems, $\varnothing', \varnothing'', \ldots$.

The relation of Turing equivalence is, naturally, an equivalence relation, and the equivalence classes are called *Turing degrees*. One speaks of the degree of a set of natural numbers to mean the equivalence class to which it belongs. The degrees inherit the partial order, and the jump operation is a degree invariant. All of the computable sets form a single degree written $\mathbf{0}$ which is at the bottom of the partial order. That is, $\mathbf{0} \leq \mathbf{a}$ for every Turing degree $\mathbf{a}$. Also, $\mathbf{a} < \mathbf{a}'$. Are there degrees between $\mathbf{a}$ and $\mathbf{a}'$? Kleene and Post were able to show that the ordering of degrees is complicated and messy. For example, for any degree $\mathbf{a}$, they showed how to obtain degrees $\mathbf{b}, \mathbf{c}$ such that $\mathbf{a} < \mathbf{b} < \mathbf{a}'$, $\mathbf{a} < \mathbf{c} < \mathbf{a}'$, but $\mathbf{b}$ and $\mathbf{c}$ are incomparable: neither is less than the other. They also found densely ordered degrees; that is, they showed that for a given degree $\mathbf{a}$, an infinite linearly ordered set $\mathcal{W}$ of degrees between $\mathbf{a}$ and $\mathbf{a}'$ can be found such that if $\mathbf{b}, \mathbf{c} \in \mathcal{W}$, there is a degree $\mathbf{d} \in \mathcal{W}$ between $\mathbf{b}$ and $\mathbf{c}$.

The degree of every recursively enumerable set is $\leq \mathbf{0}'$. There is a sense in which the typical mathematical problems that have been proved to be unsolvable are of degree $\mathbf{0}'$. For example, if we enumerate all polynomial Diophantine equations with integer coefficients in some standard way, the degree of the set of natural numbers n such that the nth equation has a solution in natural numbers is exactly $\mathbf{0}'$. So we can say that Hilbert's tenth problem is not only unsolvable but has exactly the degree of unsolvability $\mathbf{0}'$. A degree is called recursively enumerable if it contains a recursively enumerable set. $\mathbf{0}$ and $\mathbf{0}'$ are both recursively enumerable degrees with $\mathbf{0} < \mathbf{0}'$. In a classic paper Post raised the question of the existence of other recursively enumerable degrees, and this became known as Post's Problem. It required a new combinatorial technique, known as the priority method, to settle the question. The idea was to list a countable infinity of requirements that the desired objects would need to satisfy, and to mediate among conflicting requirements in a manner that would result in all of them being ultimately satisfied. By using this technique, it was shown that not only are there recursively enumerable degrees strictly between $\mathbf{0}$ and $\mathbf{0}'$, but indeed that pairs of such degrees can be found that are mutually incomparable. The use and refinement of the priority method has made it possible to prove a number of striking facts about the recursively enumerable degrees. For example, the Sacks Density Theorem states that for given recursively enumerable degrees $\mathbf{a} < \mathbf{b}$, there is a recursively enumerable degree $\mathbf{c}$ such that $\mathbf{a} < \mathbf{c} < \mathbf{b}$.

References

[1] HARTLEY ROGERS, *Theory of recursive functions and effective computability*, New York: McGraw-Hill 1967.

[2] JOSEPH SHOENFIELD, *Recursion theory*, Lecture Notes in Logic 1, Berlin, New York: Springer-Verlag, 1993.

[3] ROBERT SOARE, *Recursively enumerable sets and degrees: a study of computable functions and computably generated sets*, Berlin, New York: Springer-Verlag, 1987.