Computability and Incomputability

Robert I. Soare

Abstract

The conventional wisdom presented in most computability books and historical papers is that there were several researchers in the early 1930's working on various precise definitions and demonstrations of a function specified by a finite procedure and that they should all share approximately equal credit. This is incorrect. It was Turing *alone* who achieved the characterization, in the opinion of Gödel. We also explore Turing's oracle machine and its analogous properties in analysis.

Keywords: Turing a-machine, computability, Church-Turing Thesis, Kurt Gödel, Alan Turing, Turing o-machine, computable approximations, effectively continuous functions on reals, computability in analysis, strong reducibilities reexamined.

1 The Modern Era of Computability Theory

Mathematicians have studied algorithms and calculation at least since the time of the Babylonians and later Euclid (c. 330 B.C.). However, it was only in the modern period which began in the 1930's that mathematicians were able to give precise formal models which characterized the informal notion of a finite procedure, and harness these models in what evolved into modern computers. The modern period of the theory of computability can be split into three periods.

1. **λ-Definability Era: 1931–1935**
2. **Recursion Theory Era: 1935–1995**
3. **Computability Era: 1996-present**

1.1 Transition From λ-Definable to Recursive in 1935

Kleene arrived as a graduate student of Church in Princeton in 1935 and worked on showing that a large class of number theoretic functions were

λ-definable. Kleene gave lectures to audiences of mathematicians but was disappointed that they were unfamiliar with the λ-calculus definitions and failed to appreciate Kleene's work.

In the spring of 1934 Gödel moved to Princeton and gave his lectures on the "general recursive functions." Both Kleene and Church immediately switched from the λ-definable formalism to the Herbrand-Gödel general recursive functions. Church introduced the use of "recursive" as an adverb to mean "computable," e.g. "recursively enumerable," and Kleene later introduced the term "recursive function theory." The term "recursive" had previously meant "inductive," but in 1935 under Church and Kleene, it acquired the meaning "computable" for the sake of explaining the results to mathematicians unfamiliar with the λ-definable terminology.

1.2 Transition From Recursive to Computable in 1996

By 1995 the computer revolution had brought computers and their concepts and terminology into everyday lives. One of the biggest influences was the introduction in 1981 of the IBM personal computer (PC) which meant that more and more ordinary people had a PC on their desks by 1995. Not only scientists but the general population recognized the basic concepts and terminology of computing, but very few recognized the terminology of "recursive." Those who did associated it with "inductive" not "computable."

Soare wrote a paper *Computability and Recursion* [1996] whose content was delivered in an invited address at the International Congress for Logic, Methodology and Philosophy of Science in Florence in August, 1995. He pointed out that in the 1930's the principal founders of Computability Theory, Turing and Gödel, *never* used the term "recursive" to mean computable and explicitly rejected such suggestions. The field was ripe for a change of terminology to make itself better understood by the public, just as Kleene and Church had changed terminology. Soare's paper suggested that using the term "computable" would not only be more recognizable by the public, but would be more scientifically and historically accurate. By the time the American Mathematical Society international meeting on the subject was held in Boulder, Colorado in 1999 (see Soare [2000]) the majority of the researchers had changed to the computability terminology, and the title of the conference was now *Computability Theory and its Applications: Current Trends and Open Problems.*

1.3 Three Main Points in [1996]

Soare's paper [1996] on computability was not by itself responsible for the change which took place from 1996 to 1999 with "recursive" replaced by "computable." The seeds of change were already there. However, Soare's paper made three main points which have been subsequently confirmed.

1. The subject is about *computability*, not recursion, not λ-definability, not Post canonical systems. The subject is about studying functions which can be "computed by a finite procedure," to use the words of Gödel [1934].

2. It was *Turing*, not Kleene, not Church, not Post, not even Gödel. It was Turing *alone* who: (1) gave the first convincing formal definition of a computable function (Turing a-machine); (2) proved that the informal notion coincided with this formal one; (3) defined the *universal* Turing machine; and (4) defined the Turing oracle machine (o-machine), the central concept in computability.

3. Finally, if we use terminology and notation of computability as Turing did, and indicate our interest in wider questions beyond narrow technical ones as Turing did, then we can form connections with other diverse researchers also interested in other aspects of computability.

This has indeed happened since 1996. An organization called *Computability and Complexity in Analysis (CCA)* has held several meetings, the most recent in November, 2006, in Gainesville, Florida. A related organization, *Computability in Europe (CIE),* is sponsoring this conference and this paper. Both have attracted a large, diverse collection of members. The lectures reflect a much wider range of interests in computability than in recursion theory a decade or so ago. Could either organization have attracted so many members today with the term *"recursion"* in place of *"computability"* in the title?

2 Defining Computability in the 1930's

2.1 Several Formalisms Emerge

Gödel's Incompleteness Theorem [1931] not only solved the first Hilbert question on whether a formal system could prove its own consistency, but it generated a lot of interest in Hilbert's second question, the *Entscheidungsproblem*, decision problem, described in Hilbert and Ackermann [1928]

for first order logic. Hilbert had characterized this as the fundamental problem of mathematical logic. In Princeton Alonzo Church had introduced a precise formal system called the λ-*calculus* (now used as a programming language), but by 1931 he knew only that the successor function and addition were λ-definable. Stephen Kleene arrived in Princeton in 1931 as a student of Church, and Church put him to work adding to the knowledge of λ-definable functions. The first version of Church's λ-notation was shown to be inconsistent by Rosser and Kleene, but Church and Kleene focused on a more restricted version now known as the λ-*calculus.*. By 1934 Kleene had shown that virtually all common functions in number theory algebra were λ-definable.

Gödel, by then the most famous and respected mathematical logician in the world, moved to Princeton in 1934. In [1931] he had introduced two elements which would play an important role in computability: the use of primitive recursive functions to code configurations; and the use of sequences to code a sequence of syntactical objects such as a proof. However, Gödel knew that the primitive recursive functions did not exhaust all the computable functions. In the spring of 1934 Gödel [1934] gave a series of lectures, recorded by Kleene and Rosser, modifying an idea of Herbrand to produce what Gödel called a *general recursive function* to distinguish it from the 1931 functions he had called "recursive function" (*"rekursiv"* in German). By 1936 Gödel's terminology was changed by Church and Kleene so that "recursive function" came to mean "general recursive function" and the adjective "primitive" was added to the earlier notion.

2.2 Towards a Computability Thesis

The history of the development of a thesis characterizing the computable functions is fascinating. Gödel [1934] wrote:

2.2.1 Gödel Considers a Thesis in [1934]

> "[Primitive] recursive functions have the important property that, for each given set of values for the arguments, the value of the function can be computed by a finite procedure[3]."
>
> *Footnote 3.*
> "The converse seems to be true, if, besides recursion according to scheme (V) [primitive recursion], recursions of other forms (e.g., with respect to two variables simultaneously) are admitted. This

cannot be proved, since the notion of finite computation is not defined, but it serves as a heuristic principle."

We shall return to this quote, especially the second paragraph (his footnote 3), which gives us crucial insight into Gödel's thinking about the computability thesis and his later pronouncements about the achievements of Turing versus others.

2.2.2 Church's Thesis "Thoroughly Unsatisfactory"

Largely on the basis of the evidence of the large class of number theoretic functions shown to be λ-definable by Kleene, and based also on his own intuition, Church proposed privately to Gödel the first version of Church's Thesis. Around March, 1934, Church suggested to Gödel that the notion of "λ-definable" be identified with "effectively calculable" (which was Church's term for what we now call "intuitively computable.")

Gödel strongly rejected this suggestion of Church which he called "thoroughly unsatisfactory." Undeterred by this encounter with Gödel, Church changed formal definitions from "λ-definable" to the (Herbrand-Gödel) (general) *recursive* functions just introduced by Gödel in his lectures in 1934. This time without consulting Gödel, Church presented on to the American Mathematical Society on April 19, 1935, his famous proposition published in *1936* and known since Kleene [1952] as *Church's Thesis.*

Thesis 3. (Church's Thesis) *[1935] and [1936]. "In this paper a definition of* recursive function of positive integers *which is essentially Gödel's is adopted. It is maintained that the notion of an effectively calculable function of positive integers should be should be identified with that of a recursive function, . . . "*

Church always presented this as a *definition* of an effectively calculable function, not as a thesis. It was Kleene who much later called it a *Thesis* [1943] and finally [1952] called it "Church's Thesis." Using the identification of solvable problems with those solved by a recursive function, Church went on to exhibit an unsolvable problem in mathematics, and hence a negative solution to Hilbert's *Entscheidungsproblem.*

3.1 Church and Kleene Collect Evidence

By the beginning of 1936 Church had become ever more confident of his formal definition of effectively calculable function and its application to unsolvable problems. He had announced his work [1935] to the American

Mathematical Society. Church's major paper [1936] had been submitted and would soon appear on recursive functions and the definition of effectively calculable functions. In an abstract received by the American Mathematical Society in July, 1935, which would soon appear as Kleene [1936b], Kleene had announced the equivalence of the formal definitions of λ-definable and recursive, which Kleene and Church had shown.

Kleene had introduced a third formalism, the *μ-recursive functions,* the least class of functions closed under the former schemes (I)–(V) for the primitive recursive functions and also scheme (VI) the least number operator, $\psi(x) = (\mu y)[\theta(x, y) = 0]$. This definition was derived (as Kleene pointed out) from Gödel's [1931] paper with its use of coding of syntax using primitive recursive functions. Also received on July 1, 1935, was an abstract of the paper to appear as Kleene [1936] in which Kleene stated that every (general) recursive function is μ-recursive. This gives a useful and succinct formalism, but it is entirely derivative of Gödel [1931] because its two key ingredients are the primitive recursive functions used for coding as in Gödel and the idea of coding of syntax (or here coding steps in the computation).

The coincidence of these formal definitions of effectively calculable function gave Church more confidence that he had indeed correctly captured the informal notion of finite procedure. By January, 1936 Church had already written this for his major paper Church [1936] to appear shortly.

> "The fact, however, that two such widely different and (in the opinion of the author) equally natural definitions of effective calculability turn out to be equivalent adds to the strength of the reasons addressed below for believing that they constitute as general a characterization of this notion as is consistent with the usual intuitive understanding of it."

3.2 Stalemate at Princeton in Early 1936

Church had accumulated more and more evidence for his case, but he still did not have the approval of one man who counted most to him and to the community in mathematical logic. Gödel continued to reject Church's Thesis. He was not convinced by the confluence argument above even though he himself had supplied virtually all the essential mathematical ingredients in [1931] and [1934] for the work by Church and Kleene including the (general) recursive functions, and the use of primitive recursive functions and coding of sequences of syntactical objects later used by Kleene in the μ-recursive functions and T predicate and normal form. Why did Gödel reject Church's

Thesis by early 1936, and why did he not put forward a thesis himself in 1934?

3.2.1 Gödel's Doubts

First, Gödel had hinted in footnote 3 of [1934] quoted above in S2.2.1 that if other recursions were added they might comprise all mechanically calculable functions. However, in a letter to Martin Davis dated February 15, 1965, Gödel wrote as follows.

> ... "It is *not true* that footnote 3 is a statement of Church's Thesis. The conjecture stated there refers to the equivalence of "finite (computation) procedure" and "recursive procedure." However, I was, at the time of these lectures, not at all convinced that my concept of recursion comprises all possible recursions ..." *-Gödel 1965, letter to Martin Davis*

Second, Gödel had written in footnote 3,

> "This cannot be proved, since the notion of finite computation is not defined, but it serves as a heuristic principle."

For Gödel a crucial ingredient was to analyze the *intrinsic nature* of the notion of "finite procedure" not just prove the confluence of various formal definitions as Church and Kleene had done. In January, 1936 Gödel not only believed that this had not *yet* been done, but he also expressed some doubt as to whether "finite procedure" *could* be formally analyzed at all or whether it must serve only "as a heuristic principle."

4 Turing Breaks the Stalemate

Those gathered at Princeton, Gödel, Church, Kleene, Rosser, and Post nearby, constituted the most distinguished and powerful group of scholars in the world working on the notion of a computable function, and yet as the year 1936 began they could not agree. At that moment stepped forward a twenty-two year old youth. Well, not just any youth. Alan Turing had already proved the Central Limit Theorem in probability theory (not knowing it had been previously proved, see Zabell [1995]), and as a result Turing had been elected a Fellow of King's College, Cambridge.

The work of Hilbert and Gödel had already attracted interest at Cambridge University where topologist Professor M.H.A. (Max) Newman gave

lectures on Hilbert's *Entscheidungsproblem* in 1935. Alan Turing attended. Turing's mother had a typewriter which fascinated him as a boy. He designed his *automatic machine (a-machine)* as a kind of typewriter with an infinite carriage over which the reading head passes with the ability to read, write, and erase one square at a time. Equally important with this Turing machine was Turing's analysis of the intuitive conception of a "function produced by a mechanical procedure." In a masterful demonstration, which Robin Gandy considered as precise as most mathematical proofs, Turing analyzed the informal nature of functions computable by a finite procedure, and demonstrated that they coincide with those computable by an a-machine. Also Turing [1936, p. 243] introduced the universal Turing machine which has great theoretical and practical importance.

Thesis 5. (Turing's Thesis [1936]) *A function on the integers is computable by a finite procedure if and only if it is computable by a Turing a-machine.*

Church [1936] had tried to give a similar informal argument for Church's Thesis but Gandy [1988, p. 79] and especially Sieg [1994, pp. 80, 87] in their excellent analyses brought out the weakness in Church's argument. In 1936 Turing's was the only convincing demonstration of any thesis that a formal definition captured the informal notion of computable.

6 Gödel's Opinion of Turing's Work

Gödel's reaction was swift and emphatic. He never accepted Church's Thesis, but he accepted Turing's Thesis at once.[1]

> "That this really is the correct definition of mechanical computability was established beyond any doubt by Turing."
> -*Gödel* 193? *Notes in Nachlass [*1935*]*

> " But I was completely convinced only by Turing's paper."
> -*Gödel: letter to Kreisel of May 1, 1968 [Sieg, 1994, p. 88].*

[1] Gödel was interested in the *intensional* analysis of finite procedure. He never believed the arguments and confluence evidence which Church presented to justify his Thesis. On the other hand Gödel accepted immediately not only Turing machines, but more importantly the analysis Turing gave of a finite procedure. The fact that Turing machines were later proved *extensionally* equivalent to general recursive functions did not convince Gödel of the intrinsic merit of the other definitions.

> " ... one [Turing] has for the first time succeeded in giving an absolute definition of an interesting epistemological notion, *i.e.*, one not depending on the formalism chosen."
>
> *-Gödel, Princeton Bicentennial, [1946, p. 84].*

> ... For the concept of computability, however, although it is merely a special kind of demonstrability or decidability, the situation is different. By a kind of miracle it is not necessary to distinguish orders, and the diagonal procedure does not lead outside the defined notion.
>
> *—Gödel: [1946, p. 84], Princeton Bicentennial*

> "The greatest improvement was made possible through the precise definition of the concept of finite procedure, ... This concept, ... is equivalent to the concept of a 'computable function of integers' ... The most satisfactory way, in my opinion, is that of reducing the concept of finite procedure to that of a machine with a finite number of parts, as has been done by the British mathematician Turing."
>
> *—-Gödel [1951, pp. 304–305], Gibbs lecture*

> " ... due to A.M. Turing's work a precise and unquestionably adequate definition of the general concept of formal system can now be given, the existence of undecidable arithmetical propositions and the non-demonstrability of the consistency of a system in the same system can now be proved rigorously for *every* consistent formal system containing a certain amount of finitary number theory."
>
> *-Godel's Postscriptum to Gödel [1934], see Davis, [1965].*

6.1 Gödel on Church's Thesis

In June, 1964, Gödel remarked (see Davis [1965, p. 72]).

> "See A. Turing [1936] and the almost simultaneous paper by E.L. Post [1936]. As for previous equivalent definitions of computability, which, however, are much less suitable for our purpose, see A. Church [1936]."

6.2 Kleene Said About Turing

> "Turing's computability is intrinsically persuasive" but "λ-definability is not intrinsically persuasive" and "general recursiveness scarcely

so (its author Gödel being at the time not at all persuaded)."

-Stephen Cole Kleene [1981b, p. 49]

"Turing's machine concept arises from a direct effort to analyze computation procedures as we know them intuitively into elementary operations. Turing argued that repetitions of his elementary operations would suffice or any possible computation. For this reason, Turing computability suggests the thesis more immediately than the other equivalent notions and so we choose it for our exposition."

-Stephen Cole Kleene, second book [1967, p. 233]

6.3 Church Said About Turing

Computability by a Turing machine, " has the advantage of making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately—*i.e.*, without the necessity of proving preliminary theorems."

-Alonzo Church, [1937], Review of Turing [1936]

7 Turing Defines Oracle Machines

After Turing's discovery in April, 1936, Professor Newman suggested that he go to Princeton to take his Ph.D., which he did under Church. Turing's thesis in 1939 was on ordinal logics, an attempt to get around Gödel's incompleteness theorem by adding new axioms. In an obscure part of his paper Turing [1939, §4] wrote,

> "Let us suppose we are supplied with some unspecified means of solving number-theoretic problems; a kind of oracle as it were. ... this oracle ... cannot be a machine.
>
> 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."

There are several ways that a Turing machine with oracle may be defined. We prefer the definition in Soare [1987, p. 46] of a machine with a head which reads the work tape and oracle tape simultaneously. Any of these definitions gives the definition of a Turing functional $\Phi_e^A(x) = y$. The crucial point is that any definition must produce a computably enumerable (c.e.) set V_e as

the *graph*[2] of Φ_e,

$$V_e \;=\; \{\langle \sigma, x, y\rangle \;:\; \Phi_e^{\sigma}(x) = y\}.$$

These Turing computable functionals are exactly like the more general continuous functions on Cantor space 2^{ω} but relativized to an oracle set $X \subset om$. A continuous function on Cantor space is one with graph V_e^X for some $x \subset \omega$. Is a a researcher in analysis wants to work on the slightly different space of the real numbers with the usual topology in analysis given by open balls B_i with rational center and rational radius as the basic open sets, then the description is exactly analogous. If the function is continuous, then the inverse of a basic open set B_i is set of balls c.e. in X for some read $X \subset \omega$.

In September, 1939, Turing entered the world of British crypotography and did not develop the notion of oracle machine further. Post [1944] began to explore the notion of oracle machines and their associated definition of Turing reducibility $B \leq_{\mathrm{T}} A$, and Post studied a number of stronger less general reducibilities, such as 1-reducible, m-reducible, *btt*-reducible, *tt*-reducible (truth table reducible). The full Turing reduciblity was not well understood until at least a decade later with the Kleene Post [1954] paper finding Turing incomparable sets below $\emptyset'$.

Many problems in the real world are not explicitly computable but are limit computable. The function f may be represented as $f(x) = \lim_s \widehat{f}(x, s)$ for some computable function $\widehat{f}(x, s)$. These problems in turn can be looked at via oracle machines. Consider the halting problem $K = \{\, e : e \in W_e \,\}$.

Lemma 7.1. [Limit Lemma] *A is limit computable iff $A \leq_{\mathrm{T}} K$.*

Many processes can be looked at as having a fixed finite control, but an oracle which may be changed as external conditions warrant, *i.e.*, an oracle K_s at stage s which goes to a limit $K = \cup_s K_s$. Thus, the concept of Turing's oracle machine (o-machine) is the most important in the subject of computability at both the theoretical and practical level.

A number of textbooks on computability theory follow Post's lead by defining a Turing a-machine in chapter 1 and not defining a Turing o-machine and Turing functionals until a much later chapter, spending the intermediate chapters on the intermediate reducibilities or other smaller themes. This is analogous to having a calculus textbook delay the definition of continuous or differentiable function until chapter 10 spending the time on derivates of special functions like polynomials. We should proceed as quickly to the full definition of a Turing functional Φ_e^A and its graph V_e defined above.

[2]The term "graph" for this set is becoming standard by analogy with the graph of partial computable (p.c.) function φ_e which is the same but with out the σ.

References

[1] [Church, *1936*] A. Church, An unsolvable problem of elementary number theory, *American J. of Math.*, **58** (1936), 345-363.

[2] [Church, *1937*] A. Church, Review of Turing *1936*, *J. Symbolic Logic* **2(1)** (1937), 42–43.

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

[4] [Davis, *1982*] M. Davis, Why Gödel did not have Church's Thesis, *Information and Control* **54** (1982), 3–24.

[5] [Gandy, *1980*] R. Gandy, Church's thesis and principles for mechanisms, In: *The Kleene Symposium*, North-Holland, (1980), 123–148.

[6] [Gandy, *1988*] R. Gandy, The confluence of ideas in 1936, In: Herken, *1988*, 55–111.

[7] [Gödel, *1931*] K. Gödel, Über formal unentscheidbare sätze der Principia Mathematica und verwandter systeme. I, *Monatsch. Math. Phys.* **38** (1931) 173-178. (English trans. in Davis *1965*, 4–38, and in van Heijenoort, 1967, 592–616.

[8] [Gödel, *1934*] K. Gödel, On undecidable propositions of formal mathematical systems, Notes by S. C. Kleene and J. B. Rosser on lectures at the Institute for Advanced Study, Princeton, New Jersey, 1934, 30 pp. (Reprinted in Davis *1965* [3, 39–74]).

[9] [Gödel, *193?*] K. Gödel, Undecidable diophantine propositions, In: Gödel *1995*, 156–175.

[10] [Gödel, *1946*] K. Gödel, Remarks before the Princeton bicentennial conference of problems in mathematics, 1946. Reprinted in: Davis *1965* [3], pp. 84–88.

[11] [Gödel, *1951*] K. Gödel, Some basic theorems on the foundations of mathematics and their implications, In: Gödel *1995*, 304–323. (This was the Gibbs Lecture delivered by Gödel on December 26, 1951 to the Amer. Math. Soc.)

[12] [Gödel, *1964*] K. Gödel, Postscriptum to Gödel *1931*, written in 1946, printed in Davis *1965*, 71–73.

[13] [Hilbert and Ackermann, *1928*] D. Hilbert and W. Ackermann, *Grundzüge der theoretischen Logik*, Springer, Berlin, 1928 (English translation of 1938 edition, Chelsea, New York, 1950).

[14] [Hodges, *1983*] A. Hodges, *Alan Turing: The Enigma*, Burnett Books and Hutchinson, London, and Simon and Schuster, New York, 1983.

[15] [Kleene, *1936*] S. C. Kleene, General recursive functions of natural numbers, *Math. Ann.* **112** (1936), 727–742.

[16] [Kleene, *1943*] S. C. Kleene, Recursive predicates and quantifiers, *Trans. A.M.S.* **53** (1943), 41–73.

[17] [Kleene, *1952*] S. C. Kleene, *Introduction to Metamathematics*, Van Nostrand, New York (1952). Ninth reprint 1988, Walters-Noordhoff Publishing Co., Groningën and North-Holland, Amsterdam.

[18] [Kleene, *1967*] S. C. Kleene, *Mathematical Logic*, John Wiley and Sons, Inc., New York, London, Sydney, 1967.

[19] [Kleene, *1981*] S. C. Kleene, Origins of recursive function theory, *Annals of the History of Computing*, **3** (1981), 52–67.

[20] [Kleene, *1987*] S. C. Kleene, Reflections on Church's Thesis, *Notre Dame Journal of Formal Logic,* **28** (1987), 490–498.

[21] [Kleene, *1988*] S. C. Kleene, Turing's analysis of computability, and major applications of it, In: Herken *1988*, 17–54.

[22] [Kleene-Post, *1954*] S. C. Kleene and E. L. Post, The upper semilattice of degrees of recursive unsolvability, *Ann. of Math.* **59** (1954), 379–407.

[23] [Post, *1936*] E. L. Post, Finite combinatory processes–formulation, *J. Symbolic Logic* **1** (1936) 103–105. Reprinted in Davis *1965*, 288–291.

[24] [Post, *1944*] E. L. Post, Recursively enumerable sets of positive integers and their decision problems, *Bull. Amer. Math. Soc.* **50** (1944), 284–316. Reprinted in Davis *1965*, 304–337.

[25] [Sieg, *1994*] W. Sieg, Mechanical procedures and mathematical experience, In: A. George (ed.), *Mathematics and Mind*, Oxford Univ. Press, 1994.

[26] [Soare, *1987*] R. I. Soare, *Recursively Enumerable Sets and Degrees: A Study of Computable Functions and Computably Generated Sets*, Springer-Verlag, Heidelberg, 1987.

[27] [Soare, *1996*] R. I. Soare, Computability and recursion, *Bulletin of Symbolic Logic* **2** (1996), 284–321.

[28] [Soare, 2000] R. I. Soare, Extensions, Automorphisms, and Definability, in: P. Cholak, S. Lempp, M. Lerman, and R. Shore, (eds.) Computability Theory and its Applications: Current Trends and Open Problems, American Mathematical Society, Contemporary Math. #257, American Mathematical Society, Providence, RI, 2000. pps. 279–307.

[29] [Soare, cta] R. I. Soare, *Computability Theory and Applications,* Springer-Verlag, Heidelberg, to appear.

[30] [Turing, *1936*] A. M. Turing, On computable numbers, with an application to the Entscheidungsproblem, *Proc. London Math. Soc.* ser. 2 **42** (Parts 3 and 4) (1936) 230–265; [Turing, *1937*] A correction, ibid. **43** (1937), 544–546.

[31] [Turing, *1939*] A. M. Turing, Systems of logic based on ordinals, *Proc. London Math. Soc.* **45** Part 3 (1939), 161–228; reprinted in Davis [1965], 154–222.

[32] [Zabell, *1995*] S. L. Zabell, Alan Turing and the Central Limit Theorem, *American Mathematical Monthly* **102** No. 6 (Jun.-Jul. 1995), 483–494.

Department of Mathematics
University of Chicago
Chicago, Illinois 60637-1546
soare@uchicago.edu
URL http://www.people.cs.uchicago.edu/∼soare/