CALCULATIONS BY MAN AND MACHINE: CONCEPTUAL ANALYSIS*

WILFRIED SIEG

§1. **Analysis & history.** To investigate calculations is to analyze symbolic processes carried out by calculators; that is a lesson we owe to Turing. Taking the lesson seriously, I will formulate restrictive conditions and well-motivated axioms for two types of calculators, namely, for *human (computing) agents* and *mechanical (computing) devices*.[1] My objective is to resolve central foundational problems in logic and cognitive science that require a deeper understanding of the nature of calculations. Without such an understanding, neither the scope of undecidability and incompleteness results in logic nor the significance of computational models in cognitive science can be explored in their proper generality. The claim for logic is almost trivial and implies the claim for cognitive science; after all, the relevant logical notions have been used when striving to create artificial intelligence or to model mental processes in humans.

The foundational problems come to the fore in arguments for Church's or Turing's Thesis, asserting that an informal notion of effective calculability is captured fully by a particular precise mathematical concept. Church's Thesis, for example, claims in its original form that the effectively calculable number theoretic functions are exactly those functions whose values are computable in Gödel's equational calculus. My strategy, when arguing for the adequacy of a notion, is to bypass theses altogether and avoid the fruitless discussion of their (un-)provability. This can be achieved by *conceptual analysis,* i.e., by sharpening the informal notion, formulating its general features axiomatically, and investigating the axiomatic framework. Such an analysis will be provided for the two types of calculators I mentioned, examining closely and recasting thoroughly work of Turing and Gandy. My paper builds on systematic and historical work I have

*This paper is dedicated to my teacher, colleague, collaborator, and friend Solomon Feferman on the occasion of his 70th birthday.

[1]I presented versions of this paper at the *Feferfest* at Stanford University on December 13, 1998, at the *Séminaire Philosophie des mathématiques et de l'informatique* at the Sorbonne on June 7, 1999, at the *Workshop on Logic and Physics of Computation* in Erice (Sicily) on July 29, 1999, and at the *11th International Congress of Logic, Methodology and Philosophy of Science* in Cracow on August 21, 1999. Thanks to Guglielmo Tamburrini and two anonymous referees for helpful suggestions that clarified some issues, and also to Dawn McLaughlin for turning an awkward Word document into a beautifully LaTeXed paper.

Reflections on the foundations of mathematics
Essays in honor of Solomon Feferman
Edited by Wilfried Sieg, Richard Sommer, and Carolyn Talcott
© 2002, Association for Symbolic Logic

pursued for more than a decade, much of it in collaboration with John Byrnes, Daniele Mundici, and Guglielmo Tamburrini. The considerations presented here reshape and extend the earlier systematic work in a novel and, for me, unexpected way; their aim is nevertheless extremely classical, namely, to provide what Hilbert called *eine Tieferlegung der Fundamente*. It will also become evident that they are embedded in an illuminating historical context.

There is general agreement that Turing, in 1936, gave the most convincing analysis of *effective calculability* in his paper "On computable numbers - with an application to the Entscheidungsproblem". It can be argued that he gave the only convincing analysis and, in addition, that the nature of his analysis is still not clearly recognized. Thus, it seems worthwhile to go back to the discussion of the mid-thirties of the last century. Section 2 sets the stage by dealing with effective calculability of number theoretic functions as investigated by Gödel, Church, and Hilbert & Bernays, whereas section 3 makes explicit Turing's truly distinctive contribution and focuses on calculations by *mechanical man* (or computors).[2] Evident limitations of the human sensory apparatus motivate boundedness and locality conditions for computors, and broad physical limitations lead to similar requirements for mechanical devices in section 4. There I adapt Gandy's work on machine computability and treat calculations by (parallel) *machines*.[3] Shifting the methodological perspective and drawing consequences from these investigations, section 5 gives axiomatic characterizations of *Turing Computors* and *Gandy Machines*. For both notions suitable *representation theorems* can be proved. The theorems guarantee that the computations of any model of the axioms can be simulated by a standard Turing machine over a two-letter alphabet.

The detailed conceptual analysis of effective calculabilty yields rigorous characterizations that dispense with theses, reveal human and machine calculability as axiomatically given mathematical concepts, and allow their systematic reduction to Turing computability.

§2. **Effective calculations.** Church reviewed Turing's "On computable numbers" a few months after its publication for the Journal of Symbolic Logic. He contrasted Turing's notion for effective calculability (via idealized machines) with his own (via λ -definability) and with Gödel's (via the equational calculus). "Of these [notions]," Church remarked, "the first has the advantage of making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately ... " Neither in this review nor anywhere else did Church give reasons, why the identification *is* immediately evident for Turing's notion, and why it *is not* for the others. In contrast, Gödel seemed to capture essential aspects of Turing's considerations when making a brief and enigmatic remark in the 1964

[2]For a detailed historical and systematic examination with extensive references to the literature see my papers (1994) and (1997), but also (Davis 1982), (Gandy 1988), and (Soare 1996).

[3]Mathematical details for the characterization of machines are presented in a companion paper, (Sieg 2000).

postscript to the Princeton Lectures he had delivered thirty years earlier: "Turing's work gives an analysis of the concept of 'mechanical procedure' ... This concept is *shown* to be equivalent with that of a 'Turing machine'."[4] But neither in this postscript nor in other writings did Gödel indicate the nature of Turing's analysis or give a proof showing that the analyzed concept is indeed equivalent to that of a Turing machine.

Gödel underlined the significance of Turing's analysis, repeatedly and emphatically. He claimed in 1964, for example, that only Turing's work provided "a precise and unquestionably adequate definition of the general concept of formal system". A formal system is defined to be a mechanical procedure for producing theorems: consequently, the adequacy of this definition rests squarely on the correctness of Turing's analysis of mechanical procedures. Gödel had an abiding interest in the issue, as it was crucial for his goal to give the most general mathematical formulation and the broadest philosophical interpretation of his incompleteness theorems. A general concept of formal system was needed to achieve that goal. Gödel himself had tried to arrive at such a concept in a quite different way, namely, by a direct characterization of effectively calculable number theoretic functions. As a first step towards such a characterization, Gödel had introduced in his Princeton Lectures *general recursive functions* via his equational calculus.[5] I will review briefly the crucial features of Gödel's definition.

The general recursive functions are taken by Gödel to be those number theoretic functions whose values can be calculated from basic equations via elementary substitution rules. This is an extremely natural approach and generalizes properly the idea underlying the definition of primitive recursive functions: the new class of functions includes in addition to all primitive recursive functions also the non-primitive recursive Ackermann function. Assume, Gödel suggests, you are given a finite sequence $\psi_1, \ldots, \psi_k$ of "known" functions and a symbol ϕ for an "unknown" one. Then substitute these symbols "in one another in the most general fashions" and equate certain pairs of the resulting expressions. If the selected set of functional equations has exactly one solution, consider ϕ as (denoting) a general recursive function. Gödel attributes this proposal to Herbrand[6] and proceeds to make two restrictions: (1) the left-hand side of given equations must be of the form $\phi(\psi_{i1}(x_1, \ldots, x_n), \ldots, \psi_{il}(x_1, \ldots, x_n))$, and (2) for every l-tuple of natural numbers the value of ϕ must be "computable in a calculus". The second condition demands more precisely that for every l-tuple $k_1, \ldots, k_l$ there is exactly one m such that $\phi(k_1, \ldots, k_l) = m$ is a "derived equation". So it remains to

[4] Gödel's *Collected Works*, volume I, 369–70. The emphases are mine. In the context of this paper (and reflecting the above discussion of Church and Gödel), I consider effective and mechanical procedure as synonymous.

[5] Recall that in the contemporaneous discusssion the class of *recursive* functions consisted of those functions we now call *primitive recursive*.

[6] As we know now, Gödel misremembered: see section 2.2 and the Appendix of (Sieg 1994). The reader should compare the discussion there with that in (Wang 1974), pp. 87-89.

specify the set of *derived equations*. That is done inductively using elementary substitution rules: the basic ones are:

(A.1) Replacing all the variables of a given equation by numerals yields a derived equation;

(A.2) All true equalities $\psi_{ij}(x_1, \ldots, x_n) = m$ are derived equations.

The rules allowing steps from already obtained equations to additional ones are formulated as follows:

(R.1) Replace occurrences of $\psi_{ij}(x_1, \ldots, x_n)$ by m, if $\psi_{ij}(x_1, \ldots, x_n) = m$ is a derived equation;

(R.2) Replace occurrences of $\phi(x_1, \ldots, x_l)$ on the right-hand side of a derived equation by m, if $\phi(x_1, \ldots, x_l) = m$ is a derived equation.

Kleene analyzed, using Gödel's arithmetization technique for describing provability in the equational calculus, the general recursive functions in his (1936). He established a version of what is now called *Kleene's Normal Form Theorem*: every general recursive function can be expressed in the form $\psi(\varepsilon y.\rho(x_1, \ldots, x_n, y) = 0)$, where ψ and ρ are primitive recursive and for every n-tuple $x_1, \ldots, x_n$ there is a y such that $\rho(x_1, \ldots, x_n, y) = 0$; ε is the ε-symbol as introduced by Hilbert and used by Gödel in his 1934 Princeton Lectures. This theorem (or rather its proof) is quite remarkable: the ease with which it allows to establish equivalences of different formulations makes it plausible that *some* stable notion has been isolated; what is needed for the proof is only that the inference or computation steps are all primitive recursive. However, the question, whether *that* stable notion corresponds to the informal concept of effective calculability, has to be answered independently.

Any formal system S that is even weakly adequate for number theory will allow the computations carried out in Gödel's equational calculus. Gödel made an important observation in 1936 and tried to use it in his answer to the question just formulated: no extension of such an S by higher types, even transfinite ones, will increase the class of calculable or computable functions.[7] "Thus," he concluded, "the notion 'computable' is in a certain sense 'absolute', while almost all metamathematical notions otherwise known (for example, provable, definable, and so on) quite essentially depend upon the system adopted." At the Princeton Bicentennial Conference in 1946, Gödel stressed again the importance of the concept of general recursiveness or Turing computability and reemphasized:

> It seems to me that this importance is largely due to the fact that with this concept one has for the first time succeeded in giving an

[7] This observation made it for the first time plausible to Gödel that a stable notion had been characterized: cf. (Sieg 1994), p. 88. He pointed out, quite forcefully, in a letter to Martin Davis of February 15, 1965 that Note 3 of his 1934 Princeton Lectures did not give a formulation of Church's Thesis. Indeed, he emphasized in the letter that "at the time of these lectures, [I was] not at all convinced that my concept of recursion comprises all possible recursions; and in fact the equivalence between my definition and Kleene's ... is not quite trivial." The reference is to Kleene's 1936 paper. For further details, cf. (Davis 1982) and (Sieg 1997), pp. 159–160.

absolute definition of an interesting epistemological notion, i.e., one not depending on the formalism chosen.[8]

Indeed, the details of the formalisms (extending arithmetic) do not matter, but it is crucial that we are dealing with formalisms at all; in other words, a precise aspect of the (unexplicated) *formal* character of the extending theories has to come into play, when arguing for the absoluteness of the concept computability. Gödel did not prove that computability is an absolute concept, neither here nor in the earlier paper. I conjecture that he used considerations similar to those underlying the proof of Kleene's Normal Form Theorem in order to convince himself of the claim in 1936. If that conjecture is correct, then Church's contemporaneous step-by-step argument for the co-extensiveness of effective calculability and general recursiveness is completely parallel.[9] Church required, when explicating effective calculability as computability in logical calculi, the inferential steps in such calculi not only to be effective, but—without any supporting reason—to be general recursive.

So we find in both these cases a hidden and semi-circular condition on "steps", a condition that allows the parallel arguments to go through. This step-condition was subsequently moved into the foreground by Hilbert & Bernays' marvelous analysis of "computations in deductive formalisms". Their analysis, presented in a supplement to the second volume of *Grundlagen der Mathematik*, used the concept of reckonable function (regelrecht auswertbare Funktion). General deductive formalisms were appropriately restricted by "recursiveness conditions", the crucial one requiring the proof predicate of such formalisms to be primitive recursive. Hilbert & Bernays showed that all reckonable functions are computable in a restricted number theoretic formalism, and that the functions computable in that formalism coincide with the general recursive ones. In this way, proper mathematical underpinnings were provided for Gödel's absoluteness claim and Church's argument, but they were provided *only relative* to the recursiveness conditions.

The conceptual work of Gödel, Church, and Hilbert & Bernays had intimate historical connections and is still of genuine and deep interest.[10] It explicated effective calculability of functions by *one core notion,* namely, computability of their values in a calculus via restricted rules. But no one gave convincing reasons

[8](CW II, p. 150); my emphasis. In the footnote added to this remark in 1965 Gödel wrote: "To be more precise: a function of integers is computable in any formal system containing arithmetic if and only if it is computable in arithmetic, where a function f is called computable in S if there is in S a computable term representing f." — Tarski's remarks at this conference, only recently published in (Sinaceur 2000), make so vivid, how important the issue of the "intuitive adequacy" of general recursiveness was taken to be.

[9]The argument was given in Church's (1936); the Thesis had been formulated publicly by Church in 1935 using Gödel's notion. — How closely related Gödel's and Church's considerations were can be seen from the letter Church wrote to the Polish logician Pepis on June 8, 1937; cf. my (1997), pp. 168–9 and Appendix A.

[10]Their work was complemented by important mathematical work of Kleene and Rosser; cf. (Sieg 1997).

for the proposed restrictions on the steps permitted in computations, i.e., all the analyses ran up against the very same stumbling block. A dramatic shift of perspective made for real progress. Instead of considering particular schemes for computing the values of number theoretic functions, Turing and Post proposed to look at underlying symbolic or combinatory processes. The shift is also dramatic in a different sense: though contiguous with the other work, it overcomes—through Turing's reflections—the stumbling block for a fundamental conceptual analysis.

§3. By mechanical man.

In his specification of *finite combinatory processes* Post used a human worker who operates in a "symbol space" and carries out, over a two-letter alphabet, exactly the kind of operations Turing machines can perform. Post thought that the support for his model would be provided inductively, by considering ever-wider formulations and reducing them to the restricted formulation I just hinted at.[11] In the argument for his model, Post did not use the fact that a human worker does the computing; Turing, in describing his model, focused so strongly on machines that some commentators have taken him to analyze machine computations. Turing examined however *human* mechanical computability and exploited, in sharp contrast to Post, limitations of the human computing agent to motivate restrictive conditions. The latter replace the unconvincing implicit or explicit recursiveness requirements in the earlier analyses. The essence of Turing's formulation is brought out by an aphoristic remark of Wittgenstein's on *Turing's 'Machines'*. "These machines," Wittgenstein said, "are humans who calculate." That is not only right, but Turing asked in the historical context in which he found himself *the* pertinent question, namely, what are the possible processes a human being can carry out (when computing a number or, equivalently, determining algorithmically the value of a number theoretic function)? The general problematic required an analysis of the idealized capacities of calculators, and precisely this focus makes the analysis epistemologically significant.[12]

In my further discussion I shall use a convention suggested by Gandy and refer by *computor* to a human computing agent who proceeds mechanically. A computor operates on certain symbolic configurations, and Turing demands *immediate recognizability* of these symbolic configurations so that the most basic computation steps need not be subdivided any further. This normative demand

[11]Cf. (Sieg 1994), pp. 91–2, and (Sieg & Byrnes 1996), section 2.

[12]In my presentation I am taking an anachronistic step by having Turing discuss "computable functions". This is mainly for expository expediency, but it is systematically justifiable as follows. Turing discusses "computable numbers", but emphasizes that he could have equally easily investigated "computable functions of an integral variable or a real or computable variable, computable predicates, and so forth". Indeed, he states emphatically: "The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique." (Turing 1936), p. 116. — That is clearly in the broad tradition of Leibniz's "Calculemus!", and it is needed for the negative resolution of the Entscheidungsproblem (and other mathematical and metamathematical issues). That some such "machine simulation" should be considered is clear from other contemporaneous remarks, e.g., in (Gödel 1933).

and the evident limitation of a computor's sensory apparatus[13] lead to a number of restrictive conditions that can be extracted from Turing's discussion.[14] To formulate them in the most straightforward way, I am using two ideas, both endorsed by Turing: (i) the elimination of internal states by "more physical counterparts" (proposed in section 9 (III) of his 1936 paper), and (ii) the representation of Turing machines as Post production systems (used by Turing for obtaining new mathematical results in 1950, but also for a wonderful informal exposition of solvable and unsolvable problems in 1953).[15]

Of course, the elimination of internal states is realized in the Post representation, as the physical counterparts of such states are joined to the ordinary symbolic configurations to form complete *instantaneous descriptions* or *ids*. Any id contains exactly one such physical counterpart, and the[16] immediately recognizable sub-configuration of an id must contain it. Given this compact description, the restrictive conditions are as follows:

(B) (Boundedness) *There is a fixed bound on the number of configurations a computor can immediately recognize.*

(L) (Locality) *A computor can change only immediately recognizable (sub-) configurations.*

Computors proceed deterministically; consequently, the computing process has to satisfy:

[13]Turing goes beyond the appeal to sensory limitations and views memory limitations as the ultimate reason for the restrictive conditions; cf. (Sieg 1994), p. 96.

[14]For ease of comparison, I attach here Turing's conditions as analyzed, for example, in my (1997); a similar analysis is presented in (Wang 1974) on pp. 90–95. (However, the crucial point of grounding the boundedness and locality conditions in the limitations of the computing subject is not brought out by Wang.) A first boundedness condition can be formulated as follows:
(B.1) *There is a fixed bound on the number of symbolic configurations a computor can immediately recognize.*
A second boundedness condition concerns "internal states" reflecting the computor's experience:
(B.2) *There is a fixed bound on the number of internal states a computor can be in.*
For a given computor there are consequently only boundedly many different combinations of symbolic configurations and internal states. His behavior is taken to be deterministic, i.e., it has to satisfy a determinacy condition:
(D) *A computor's internal state together with the observed configuration fixes uniquely the next computation step and the next internal state.*
Thus, he can carry out at most finitely many different operations. The operations are restricted by locality conditions.
(L.1) *A computor can change only elements of an observed symbolic configuration.*
A second locality condition allows a computor to turn attention to a configuration "close" to the one he had been observing.
(L.2) *A computor can shift attention from one symbolic configuration to another one, but the new observed configuration must be within a bounded distance of the immediately previously observed configuration.*

[15]Turing machines were represented as production systems, of course, by Post in his superb paper "Recursive unsolvability of a problem of Thue" (1947). This representation is also used in Davis' classical textbook *Computability and Undecidability*.

[16]There is a certain conventionality to determine "the" sub-configuration at this point, as a number of different ones may be observable; cf. (Sieg & Byrnes 1996), pp. 103–6.

(D) (Determinacy) *The immediately recognizable (sub-)configuration determines uniquely the next computation step (and id).*

Turing argued that the number theoretic functions calculable by such a computor are also computable by a Turing machine. Thus, Turing's assertion that effective calculability can be identified with machine computability is the result of a two-part analysis: the first part yields the boundedness condition for symbolic configurations and the locality condition for mechanical operations; the second part argues for the *claim* that every number theoretic function calculable by a computor, satisfying these conditions, is computable by a Turing machine. As we shall see, this latter part has to be broken into two separate steps.

The above restrictive conditions on computors are given quite informally. In order to formulate them precisely and make their investigation possible, it seems that the symbolic configurations must be specified; they were taken by Turing to be linear. Here is the starting-point of Turing's considerations together with a dimension-lowering step:

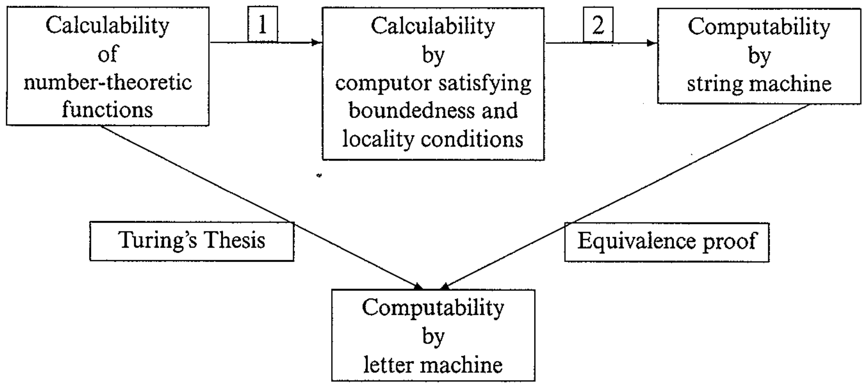
> Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child's arithmetic book. In elementary arithmetic, the two-dimensional character of the paper is sometimes used. But such a use is always avoidable, and I think that it will be agreed that the two-dimensional character of paper is no essential of computation. I assume then that the computation is carried out on one-dimensional paper, *i.e.* on a tape divided into squares.[17]

In his further reductive argument Turing constructed machines that mimic the work of computors on linear configurations. These more general machines are best seen as allowing the replacement of finite strings on the tape by finite strings. Calling such machines *string machines* and standard Turing machines *letter machines,* the assertion rigorously established by Turing is this: computations of string machines can be carried out by letter machines.[18] Turing's considerations make plausible the claim of his *Central Thesis*: computations carried out by a computor satisfying the boundedness and locality conditions can be *directly* simulated by a string machine. Computations of a string machine, in turn, can provably be mimicked by a letter machine. The diagram below represents these reflections graphically and relates them to the standard formulation of Turing's Thesis.

Step 1 in the diagram is given by conceptual analysis, whereas step 2 indicates the application of the Central Thesis. The equivalence proof justifies an extremely

[17](Turing 1936), p. 135.

[18]Turing observed: "The machines just described do not differ very essentially from computing machines as described in §2, and corresponding to any machine of this type a computing machine can be constructed to compute the same sequence, that is to say the sequence computed by the computer [in my terminology: computor]."


Calculability of number-theoretic functions
1
Calculability by computor satisfying boundedness and locality conditions
2
Computability by string machine
Turing's Thesis
Equivalence proof
Computability by letter machine

simple description of computations that is most useful for mathematical investigations, from the construction of a universal machine and the formulation of the halting problem to the proof of the undecidability of the Entscheidungsproblem. It should be underlined that step 2, not the equivalence proof, is for Turing the crucial one that goes beyond the conceptual analysis; for me it is the problematic one that requires further reflection.

In order to make Turing's Central Thesis, quite in Post's spirit, inductively more convincing, it seems sensible to allow larger classes of symbolic configurations and more general operations on them. Turing himself intended, as we saw, to give an analysis of mechanical procedures on two-dimensional configurations already in 1936. In 1953 he considered even three-dimensional configurations and mechanical operations on them, starting out with examples of puzzles: square piece puzzles, puzzles involving the separation of rigid bodies or the transformation of knots, i.e., puzzles in two and three dimensions. He viewed Post production systems as linear or *substitution puzzles*. As he considered them as puzzles in "normal form", he was able to formulate a suitable version of "Turing's Thesis":

> Given any puzzle we can find a corresponding substitution puzzle which is equivalent to it in the sense that given a solution of the one we can easily find a solution of the other ... A transformation can be carried out by the rules of the original puzzle if and only if it can be carried out by substitutions ... [19]

Turing admits that this formulation is "somewhat lacking in definiteness" and claims that it will remain so; he characterizes its status as lying between a theorem and a definition: "In so far as we know *a priori* what is a puzzle and what is not, the statement is a theorem. In so far as we do not know what puzzles are, the statement is a definition which tells us something about what they are." Of

[19](Turing 1953), p. 15.

course, Turing continues, one could define puzzle by a phrase beginning with "a set of definite rules", or one could reduce its definition to that of computable function or systematic procedure. A definition of any of these notions would provide one for puzzles. Neither in 1936 nor in 1953 did Turing characterize mathematically more general configurations and elementary operations on them. I want to describe briefly one particular attempt to do just that, by Byrnes and me in our (1996).

Our approach was influenced by Kolmogorov & Uspensky's work on algorithms and has three distinct components: the computor operates on certain finite connected and labeled graphs, we call *K(olmogorov)-graphs*; *K*-graphs contain a unique *distinguished element* that corresponds to the scanned square of a Turing machine tape; the operations *substitute neighborhoods* of the distinguished element by appropriate other neighborhoods and are given by a finite list of generalized Post production rules. Though broadening Turing's original considerations, we remain within his general analytic framework and establish that letter machines can mimic *K*-graph machines. Turing's Central Thesis is turned into the thesis that *K*-graph machines can do the work of computors directly. As a playful indication of how human algorithms can be carried out straightforwardly by *K*-graph machines, Byrnes and I programmed a *K*-graph machine to do ordinary, two-dimensional column addition. In sum, a much more general class of symbolic configurations and operations on them is considered, and the central thesis for *K*-graph machines seems even more plausible than the one for string machines.

The separation of informal conceptual analysis and mathematical equivalence proof is essential for recognizing that the correctness of *Turing's Thesis* (taken generically) rests on two pillars; namely, on the correctness of boundedness and locality conditions for computors, and on the correctness of the pertinent central thesis. The latter asserts explicitly that computations of a computor can be mimicked directly by a particular kind of machine. However satisfactory one may find this line of analytic argument, there are two weak spots: the looseness of the restrictive conditions (What are symbolic configurations? What changes can mechanical operations effect?) and the corresponding vagueness of the central thesis. We are, no matter how we turn ourselves, in a position that is methodologically still unsatisfactory. To make a step towards a more satisfactory stance, I shall first take a detour through Gandy's analysis of machine computations, abstracting further away from particular types of configurations and operations, and then return to this central methodological issue in section 5.

§4. By (parallel) machine. It has been claimed frequently that Turing analyzed computations of machines. That is historically and systematically inaccurate, as my exposition should have made quite clear. Only in 1980 did Turing's student Robin Gandy characterize machine computations. Gandy focused on "discrete mechanical devices", as opposed to possibly more general physical devices; he excluded, in particular, analogue machines from consideration. The two physical conditions characteristic for such devices are a lower bound on the size of their atomic parts (or simply "atoms") and an upper bound on the speed of signal propagation. This justifies Gandy's contention that states of such machines "can be adequately described in finite terms."[20] Calculations are taken to proceed in discrete and uniquely determined steps. Consequently, Gandy claims that these devices can be viewed, in a loose sense, as digital computers. The central and novel aspect of Gandy's analysis is the fact that it incorporates parallelism and covers cellular automata directly. This is of real interest, because cellular automata do not satisfy the locality condition (L); after all, the configurations affected in a single computation step are potentially unbounded.

Gandy's characterization—some details are presented in the Appendix—is given in terms of discrete dynamical systems $\langle \boldsymbol{S}, \boldsymbol{F} \rangle$, where $\boldsymbol{S}$ is the set of states and $\boldsymbol{F}$ governs the system's evolution. More precisely, $\boldsymbol{S}$ is a structural class, i.e., a subclass of the hereditarily finite sets $\boldsymbol{HF}$ over an infinite set $\boldsymbol{U}$ of atoms that is closed under $\in$-isomorphisms, and $\boldsymbol{F}$ is a structural operation from $\boldsymbol{S}$ to $\boldsymbol{S}$, i.e., a transformation that is, roughly speaking, invariant under permutations of atoms. These dynamical systems have to satisfy four restrictive principles. The first principle pertains to the *form of description* and states that any machine $\boldsymbol{M}$ can be presented by such a pair $\langle \boldsymbol{S}, \boldsymbol{F} \rangle$, and that $\boldsymbol{M}$'s computation, starting in an initial state $\boldsymbol{x}$, is given by the sequence $\boldsymbol{x}, \boldsymbol{F}(\boldsymbol{x}), \boldsymbol{F}(\boldsymbol{F}(\boldsymbol{x})), \ldots$. Gandy formulates three groups of substantive principles, the first of which—*The Principle of Limitation of Hierarchy*—requires that the set theoretic rank of the states is bounded, i.e., the structural class $\boldsymbol{S}$ is contained in a fixed initial segment of the $\boldsymbol{HF}$ hierarchy. Gandy argues that it is natural or convenient to think of a machine in hierarchical terms, and that "for a given machine the maximum height of its hierarchical structure must be bounded".[21] The second of the substantive principles—*The Principle of Unique Reassembly*—claims that any state can be "assembled" from "parts" of bounded size; its proper formulation requires care and a lengthy sequence of definitions. The informal idea, though, is wonderfully straightforward: any state of a concrete machine must be built up from (finitely many different types of) off-the-shelf components. Clearly, the components have a bound on their complexity. Both of these principles are concerned with the states in $\boldsymbol{S}$; the remaining third and central principle—*The Principle of Local Causality*—puts conditions on (the local determination of) the structural operation $\boldsymbol{F}$. It is formulated by Gandy in this preliminary way: "The next state, $\boldsymbol{Fx}$, of a machine can be reassembled from its restrictions to overlapping 'regions' s and these restrictions are locally caused." It requires that the parts from which $\boldsymbol{F}(\boldsymbol{x})$ can be reassembled depend only on bounded parts of $\boldsymbol{x}$.

Gandy's Central Thesis is naturally formulated as the claim that any mechanical device can be represented as a dynamical system satisfying the above principles. As to the basic set-up John Shepherdson remarked: "Although Gandy's principles

[20]Cf. (Gandy 1980) p. 126, but also pp. 135–6. For a more detailed argument see (Mundici & Sieg 1995), section 3.

[21](Gandy 1980), p. 131.

were obtained by a very natural analysis of Turing's argument they turned out to be rather complicated, involving many subsidiary definitions in their statement. In following Gandy's argument, however, one is led to the conclusion that that is in the nature of the situation."[22] — Nevertheless, in (Sieg & Byrnes 1999b) a greatly simplified presentation is achieved by choosing definitions appropriately, focusing sharply on the central informal ideas, and using one key suggestion made by Gandy in the Appendix to his paper. This simplification does not change at all the form of presentation. The subsidiary definitions are streamlined significantly, however, and of the four principles used by Gandy only a restricted version of the principle of local causality is explicitly retained. It is formulated in two separate parts, namely, as the principle of *Local Causation* and that of *Unique Assembly*. The separation reflects the distinction between the local determination of regions of the next state and their assembly into the next state. The resulting modified set-up is focused resolutely on the dynamics of processes: the static build-up of states from simple parts is no longer of concern; rather, the machines have to recognize patterns in a given state and act on them! That there are only a finite number of such patterns is justified *directly* by the two physical limitations appealed to above, namely, the lower bound on the size of atoms and the upper bound on the speed of signal propagation.

Before giving a synopsis of an even more simplified version of parallel computations, let me point to the crucial new methodological difference with Gandy's set-up. I no longer take a *Gandy machine* to be a dynamical system $\langle \boldsymbol{S}, \boldsymbol{F} \rangle$ (satisfying Gandy's principles), but rather a structure $\boldsymbol{M}$ consisting of a structural class $\boldsymbol{S}$ of states together with two kinds of patterns and operations on (instantiations of) the latter; these patterns and operations underlie also Gandy's principle of local causality. Given any dynamical system $\langle \boldsymbol{D}, \boldsymbol{F} \rangle$, where $\boldsymbol{D}$ need not be a structural class and $\boldsymbol{F}$ need not be a structural operation, $\boldsymbol{F}$ is now called *computable in parallel* if, and only if, there is a *Gandy machine* $\boldsymbol{M}$ on $\boldsymbol{S_D}$ that determines a sequence of states $z_0, z_1, z_2, \ldots$ $\in$-isomorphic to the successive states $\boldsymbol{x}, \boldsymbol{F}(\boldsymbol{x}), \boldsymbol{F}(\boldsymbol{F}(\boldsymbol{x})), \ldots$ for each $\boldsymbol{x}$ in $\boldsymbol{D}$.[23] In general, a Gandy machine $\boldsymbol{M}$ on a structural class $\boldsymbol{S}$ includes then, first of all, a finite set $\boldsymbol{T_1}$ of stereotypes, i.e., isomorphism classes of *parts* of a state $\boldsymbol{x}$, and an associated structural function $\boldsymbol{G_1}$. $\boldsymbol{G_1}$ operates on "maximal" parts or *causal neighborhoods for* $\boldsymbol{x}$, abbreviated by $Cn_1(\boldsymbol{x})$, to yield *determined regions* of some state $\boldsymbol{z}$ out of which the next state can be assembled; the regions are said to be *locally caused*. $Dr_1(\boldsymbol{z}, \boldsymbol{x})$ denotes the set of determined regions.

The first principle (**LC.1**) states that every element of $Cn_1(\boldsymbol{x})$ yields a determined region or, to put it in Gandy's words, that every cause has an effect. Indeed, the principle guarantees also that the effects are unique by requiring that determined regions of $\boldsymbol{z}$ are identical, when they are $\in$-isomorphic over $\boldsymbol{x}$; otherwise there would not be, as Gandy put it, "any bounds on the number of distinct regions which arise from a given causal neighborhood".[24] A second finite set $\boldsymbol{T_2}$ of

[22](Shepherdson 1988), p. 586.

[23]$\boldsymbol{S_D}$ is obtained from $\boldsymbol{D}$ by closing under $\in$-isomorphisms.

[24](Gandy 1980), p. 139.

stereotypes together with its associated structural operation G_2 play a significant role in the assembly of the next state and have to satisfy the analogue of (**LC.1**): every element of $Cn_2(x)$ yields a determined region in $Dr_2(z, x)$, without requiring uniqueness; this is principle (**LC.2**). The determined regions in $Dr_1(z, x)$ are the building blocks of the next state and not only cover the next state, but fit together suitably, when they *overlap*, i.e., have new atoms in common. Here T_2 and G_2 come in, as they provide local information concerning overlaps. That is expressed by principle (**GA.1**): for any collection C of regions from $Dr_1(z, x)$ with common new atoms there must be a w in $Dr_2(z, x)$, such that all elements of C are parts of w. The final principle, (**GA.2**), requires that the next state be obtained as the union of the determined regions in $Dr_1(z, x)$. A quintuple $\langle S; T_1, G_1, T_2, G_2\rangle$ is called a *Gandy Machine* M on S if and only if for every $x \in S$ there is a $z \in S$, such that the local causation conditions (**LC.1–2**) and the global assembly conditions (**GA.1–2**) are all satisfied; precise formulations are given in the Appendix.

Gandy established in his paper as the central mathematical fact that the sequence of states of such a system is Turing computable (to within $\in$-isomorphism). The proof relies on a lemma stating that, for every $x \in S$, $Dr_1(z, x)$ and z are unique up to $\in$-isomorphism over x.[25] The next state is thus computable by a Turing machine (via an exhaustive search). It may be that Gandy machines can be simplified further, for example, by a graph theoretic presentation or a category theoretic description.[26] But what is needed most, in my view, is their further mathematical investigation, e.g., for issues of complexity and speed-up, and their use in significant applications, e.g., for the analysis of DNA computations or of parallel distributed processes. The latter was done by De Pisapia in his (2000) for many important kinds of artificial neural networks.[27] The consequence is that artificial neural nets of these varieties can be simulated by Turing machines. Analogous results are obtained, using quite different techniques, by Siegelmann in (1998).

However, the most difficult and subtle aspect of Gandy machines, namely the addition of new atoms, is not used at all for these neural nets, as they have a fixed number of nodes. So there are two obvious questions: "Is there a natural subclass of Turing computable functions in which these neural nets lie?", and "Are there mental processes for whose representation this aspect of Gandy machines might

[25]The argument as given in Gandy is not quite correct; details can be found in (Sieg 2000).

[26]A graph theoretic presentation was proposed in (Byrnes & Sieg 1996). On the topic of a category theoretic definition Gandy wrote in his (1980), p. 147: "The heavy use made of restrictions ... suggests that a treatment using concepts analogous to those of sheaf theory or topos theory might be worth developing. However, it seems to me that the concepts from category theory which would be necessary would be too abstract to allow one to use them (as we have used the more concrete notions of set theory) as a justification for the main thesis of this paper." Perhaps this issue should be revisited twenty years after its original formulation. A starting-point can be found in (Herron 1995).

[27]De Pisapia considered in particular ANNs with Hebbian learning and the backpropagation algorithm. Here is a possibility for rich interaction with classical and contemporay work in the foundations of mathematics, namely, the formalization of analysis in very weak formal frameworks.

be crucial?". These are appealing and important questions, but let me turn back to the central methodological issue I set out to address.

§5. Methodological point. The analysis offered by Turing was radically new in 1936 and yet, as I argued, contiguous with the work of Gödel, Church, Hilbert & Bernays, and others. In particular, it was just a step away from Gödel's considerations in the spring of 1934: Turing analyzed in a novel and convincing way the processes underlying computations (say in Gödel's equational calculus), but—and that is crucial here—he provided also a basis for further reflections along Gödelian lines. What do I have in mind with this last remark? In a conversation with Church in early 1934, Gödel found Church's proposal to identify effective calculability with λ-definability "thoroughly unsatisfactory". As a counter-proposal he suggested "to state a set of axioms which would embody the generally accepted properties of this notion (i.e., effective calculability), and to do something on that basis".[28] Gödel did not articulate what the generally accepted properties of effective calculability might be, or what might be done on the basis of an appropriate set of axioms.

The sharpened version of Turing's work and a thorough-going re-interpretation of Gandy's approach allow us to fill in the blanks of Gödel's suggestion; this resolves, in my view, the methodological issue raised at the end of section 3. Let me bring out the central points for Gandy machines. First, the definition of a Gandy machine is an "abstract" mathematical definition that embodies generally accepted properties of parallel computations;[29] the axiomatic conditions enforce, for any Gandy machine, that the state immediately following a given state x can be obtained (uniquely up to $\in$-isomorphism over x) from the determined regions. Second, Gandy machines share with groups and topological spaces the general feature of abstract axiomatic definitions, namely, that they admit a wide variety of different interpretations.[30] Third, the central fact, that the computations of any Gandy machine can be simulated by a letter machine, is best understood as a *representation theorem* for the axiomatic notion. With these broad observations in the background, I will recast now the earlier considerations for Turing computors in such a way that they parallel those for Gandy machines.

Let $\langle D, F\rangle$ be a discrete dynamical system; the operation F is called computable if, and only if, there is a *Turing computor* M *on* S_D that determines a sequence

[28]That is reported by Church in a letter to Kleene of November 29, 1935; cf. my (1997), pp. 159–160.

[29]In (Lamport and Lynch) it is asserted that "the theory of sequential computing rests upon fundamental concepts of computability that are independent of any particular computational model." In contrast, it is claimed, no such fundamental concepts underlying distributed computing have yet been developed. That is followed by some informal observations (p. 1166): "Underlying almost all models of concurrent systems is the assumption that an execution consists of a set of discrete events, each affecting only part of the system's state. Events are grouped into processes, each process being a more or less completely sequenced set of events sharing some common locality in terms of what part of the state they affect. For a collection of autonomous processes to act as a coherent system, the processes must be synchronized."

[30]In my paper (1996) I tried to argue for the distinctive character of "abstract definitions."

of states $z_0, z_1, z_2, \ldots$ $\in$-isomorphic to the successive states $x, F(x), F(F(x)), \ldots$ for each x in D. A Turing computor M on S is given by a triple $\langle S; T, G\rangle$, where S is a structural class of states, T a finite set of patterns or stereotypes, and G the corresponding structural operation modifying instantiated patterns. Every state is required by **(C.0)** to contain exactly one observed configuration, its sole causal neighborhood, denoted by $cn(x)$. The remaining axioms for Turing computors are similar to those for Gandy machines. First, the observed configuration $cn(x)$ of state x yields a unique determined region of any possible next state z; that is expressed by principle **(LC.1)**. Second, denoting the uniquely fixed determined region of z by $dr(z, x)$, the next state is obtained as the union of $x \setminus cn(x)$ and $dr(z, x)$. This is the assembly condition **(GA.1)**. The next state z is determined uniquely up to $\in$-isomorphism over x. Looking back at the informal conditions on Turing computors, one notices readily that the boundedness and locality conditions are satisfied by M. Note also, that the principles for a Turing computor are satisfied by string and K-graph machines. A suitable representation theorem can be established, showing that computations of any model of these principles can be simulated by a letter machine.

The axiomatic approach captures the essential nature of computation processes in an abstract way. The difference between the two types of calculators I have been describing is reduced to the fact that Turing computors modify *one* bounded part of a state, whereas Gandy machines operate in parallel on *arbitrarily many* bounded parts. The representation theorems guarantee that models of the axioms are computationally equivalent to Turing machines in their letter variety. In any event, these considerations remove any appeal to theses, whether central or not. They fit the bill of Turing's appealing intuitive puzzle-approach and satisfy Gödel's demand for an axiomatically characterized notion. Indeed, they give proper content to Gödel's enigmatic remark—discussed in section 2—stating: "Turing's work gives an *analysis* of the concept of 'mechanical procedure' .... This concept is *shown* to be equivalent with that of a 'Turing machine'." As to the correctness of the analysis, an appeal to intuition in Turing's sense can no more be avoided in this case than in any other case of an axiomatically characterized mathematical structure that is intended to model broad aspects of reality.[31]

In the case under discussion this is fraught with controversy and often misunderstanding. For example, Gödel spotted a "philosophical error" in Turing's work, *assuming* that Turing's argument in the 1936 paper was to show that "mental procedures cannot go beyond mechanical procedures."[32] Not surprisingly, he

[31]The argument in section 9 of (Turing 1936)—that was analyzed above—is characterized as "a direct appeal to intuition"; Turing discusses his concept of intuition at greater length in section 11 of his 1939 paper on ordinal logics. — A second significant example of such an axiomatic analysis, "modeling broad aspects of reality", was given by Dedekind for the continuum; cf. my (1997), pp. 173–4.

[32]In his Note from 1972; that is also reported with some additional comments in (Wang 1974), pp. 324–6.

considered the argument as inconclusive. Indeed, Turing does not give a conclusive argument for Gödel's claim, but it has to be added that he did not intend to argue for it. Even in his work of the late 1940's and early 1950's that deals explicitly with mental processes, Turing does not argue that "mental procedures cannot go beyond mechanical procedures." Mechanical processes are still made precise as Turing machine computations; in contrast, machines that might exhibit intelligence have a much more complex structure.[33] Conceptual idealization and empirical adequacy are being sought now for different purposes, and Turing is trying to capture quite clearly what Gödel found missing in the analysis of (a broader concept of humanly) effective calculability, namely, "... that mind, in its use, is not static, but constantly developing." The real difference between Turing's and Gödel's views, it seems, is Gödel's belief that it is "a prejudice of our time" that "[t]here is no mind separate from matter." This is reported by Wang.[34] Gödel expected also, according to Wang, that this prejudice "will be disproved scientifically (perhaps by the fact that there aren't enough nerve cells to perform the observable operations of the mind)." Clearly, Turing did not share that expectation.

Appendix

This appendix provides some details for Gandy machines. Their states are non-empty hereditarily finite sets over an infinite set U of atoms. A class S of states is called *structural*, if S is closed under $\in$-isomorphisms. The lawlike connections between states are given by *structural operations F on S*, i.e., from S to S. Structural operations satisfy the condition:[35] for all permutations π on U and all $x \in S$, $F(x^\pi)$ is $\in$-isomorphic over x^π to $F(x^\pi)$. "x is $\in$-isomorphic over z to y" means that the $\in$-isomorphism between x and y is the identity on $\sup(z)$, i.e., the atoms in the transitive closure of z; I use the abbreviation "$x \cong_z y$".

If x is a given state, regions of the next state are *locally* determined. Thus it is important to describe suitable substructures of x on which operations can be performed. Proper subtrees y of the $\in$-tree for x are called *parts for x*, briefly $y <^* x$, if they are specified as follows:[36] $y \neq x$ and y is a non-empty subset of

$$\{v \mid (\exists z)(v <^* z \wedge z \in x)\} \cup \{r \mid r \in x\}$$

[33]Turing's speculations are described most carefully and compared thoroughly with early work by Newell and Simon in (Colvin 1997).

[34](Wang 1974), p. 326. There one finds also this elaboration: "More generally, Gödel believes that mechanism in biology is a prejudice of our time which will be disproved. In this case, one disproval, in Gödel's opinion, will consist in a mathematical theorem to the effect that the formation within geological times of a human body by the laws of physics (and any other laws of a similar nature), starting from a random distribution of the elementary particles and the field, is about as unlikely as the separation by chance of the atmosphere into its components."

[35]For motivation of this particular condition see p. 154 of (Sieg & Byrnes 1999b).

[36]I am deviating quite consciously from Gandy's terminology; my "part" (is more general than, but) corresponds roughly to "located subassembly" in (Gandy 1980) and to "subassembly" in (Sieg & Byrnes 1999b). The reader should also compare it to Gandy's $\subseteq^*$, p. 136. Gandy remarks: "If one

If the non-empty subset in this $\in$-recursive definition consists at each stage of exactly one element, y is a *path through* x. Paths through x are of the form $\{r\}^n$, for a natural number n and some atom r. A collection C of parts for x is a *cover for* x just in case for every path y through x there is a $z \in C$, such that y is a path through z.

The local operations are given by a structural operation G_1 that works on parts y for x. Each y lies in one of a finite number of isomorphism classes (or stereotypes).[37] So let T_1 be a fixed, finite class of stereotypes: a part for x that is a member of a stereotype of T_1 is called, naturally enough, a T_1 *-part for* x. A T_1 -part y for x is a *causal neighborhood for* x given by T_1, briefly $y \in Cn_1(x)$,[38] if there is no T_1 -part y^* for x such that y is $\in$-embeddable into y^*. G_1 operates on such causal neighborhoods. The values of G_1, however, are in general not exactly what is needed for the assembly of the next state. For that purpose, we introduce *determined regions* of a state z obtained from causal neighborhoods for x : $v \in Dr_1(z,x)$ if and only if $v <^* z$ and there is a $y \in Cn_1(x)$, such that $G_1(y) \cong_y v$ and $\sup(v) \cap \sup(x) \subseteq \sup(y)$. The last condition for Dr_1 guarantees that new atoms in $G_1(y)$ correspond to new atoms in v, and that the new atoms in v are new for x. If one requires G_1 to satisfy similarly $\sup(G_1(y)) \cap \sup(x) \subseteq \sup(y)$, then the condition "$G_1(y) \cong_y v$" can be strengthened to "$G_1(y) \cong_x v$". The new atoms are thus always taken from $U \setminus \sup(x)$. Note that the number of new atoms introduced by G_1 is bounded, i.e., $|\sup(G_1(y)) \setminus \sup(x)| < n$ for some natural number n (any $x \in S$ and any causal neighborhood y for x). The determined regions have to be assembled into the next state, and for that, a second structural operation G_2 and a second set T_2 of stereotypes are needed. Finally, we have all the ingredients of a Gandy Machine.

Definition. $M = \langle S; T_1, G_1, T_2, G_2\rangle$ is a *Gandy Machine* on S, where S is a structural class, T_i a finite set of stereotypes, G_i a structural operation on (the elements of) T_i, if and only if, for every $x \in S$ there is a $z \in S$, such that

(LC.1) $(\forall y \in Cn_1(x))(\exists! v \in Dr_1(z,x))\ v \cong_x G_1(y)$
(LC.2) $(\forall y \in Cn_2(x))(\exists v \in Dr_2(z,x))\ v \cong_x G_2(y)$
(GA.1) $(\forall C)[C \subseteq Dr_1(z,x) \wedge \bigcap\{\sup(v) \cap A(z,x) | v \in C\} \neq \emptyset \rightarrow (\exists w \in Dr_2(z,x))(\forall v \in C)\ v <^* w]$;[39]
(GA.2) $z = \bigcup Dr_1(z,x)$.

considers y as a tree of its $\in$-chains, then $u \subseteq^* y$ implies that u is a subtree with the same vertex as y." The relation is defined by the condition $(\exists s \subseteq Tc(y))u = y \uparrow s$.

[37]This operation is an operation on x and y, as it introduces, possibly, new atoms—new for x. It has in this very weak sense a "global" aspect; however, as it is a structural operation, the precise choice of the atoms does not matter at all.

[38]Causal neighborhoods are of course implicitly dependent on the set T_1 of stereotypes.

[39]In Gandy's set-up the finiteness of the C has to be forced axiomatically; here it is a trivial consequence of the finiteness of Dr_1. Furthermore, principle **(GA.1)** forces a fixed upper bound on the number of determined regions that have new atoms in common.

LC stands for *Local Causation*, whereas **GA** abbreviates *Global Assembly*; $A(z,x)$ abbreviates $\sup(z) \setminus \sup(x)$. By a slight abuse of language a z satisfying the conditions is denoted by $M(x)$. That is justified by the central fact established by Gandy: $Dr_1(z,x)$ and z are unique up to $\in$-isomorphism over x.[40] This is the systematic background for the next definition.

Definition. Let $\langle D, F\rangle$ be any discrete dynamical system; F is called *computable in parallel* if and only if there is a Gandy machine M on S_D, such that for each $x \in D : F(x) \cong_x M(x)$.

The second assembly condition implies that $Dr_1(z,x)$ is a cover for z. The central fact for Gandy's proof, establishing the Turing computability of the sequence of states, is formulated now in my setting as follows:

Theorem. Let M be $\langle S; T_1, G_1, T_2, G_2\rangle$ as above and $x \in S$; if there are z and z' in S satisfying principles **(LC.1)**–**(LC.2)**, **(GA.1)**, and such that $Dr_1(z,x)$ and $Dr_1(z',x)$ cover z and z', then $Dr_1(z,x) \cong_x Dr_1(z',x)$.

Now I formulate analogous conditions for Turing Computors.

Definition. $M = \langle S; T, G\rangle$ is a *Turing Computor on* S, where S is a structural class, T a finite set of stereotypes, and G a structural operation on T, if and only if, for every $x \in S$ there is a $z \in S$, such that

(LC.0) $(\exists! y) y \in Cn(x)$
(LC.1) $(\exists! v <^* z)\ v \cong_x G(cn(x))$;
(GA.1) $z = (x \setminus cn(x)) \cup dr(z,x)$.

$cn(x)$ and $dr(z,x)$ denote the sole causal neighborhood of x, respectively the determined region of z.

REFERENCES

J. Byrnes and W. Sieg [1996], *A graphical presentation of Gandy's parallel machines (abstract)*, ***The Bulletin of Symbolic Logic***, vol. 2, pp. 452–3.

A. Church [1936], *An unsolvable problem of elementary number theory*, ***American Journal of Mathematics***, vol. 58, pp. 345–363, reprinted in (Davis 1965).

A. Church [1937], *Review of (Turing 1936)*, ***The Journal of Symbolic Logic***, vol. 2, no. 1, pp. 40–41.

S. Colvin [1997], *Intelligent machinery: Turing's ideas and their relation to the work of Newell and Simon*, ***Master's thesis***, Carnegie Mellon University, Department of Philosophy, 63pp.

M. Davis [1958], ***Computability and unsolvability***, McGraw-Hill, New York.

M. Davis (editor) [1965], ***The undecidable: Basic papers on undecidable propositions, unsolvable problems, and computable functions***, Raven Press, Hewlett, New York.

[40]In (Gandy 1980) this uniqueness up to $\in$-isomorphism over x is achieved in a much more complex way, mainly, because parts of a state are proper subtrees, in general non-located. Given an appropriate definition of cover, a collection C is called an *assembly for* x, if C is a cover for x and the elements of C are maximal. The fact that C is an assembly for exactly one x, if indeed it is, is expressed by saying that C *uniquely assembles to* x. Indeed, in my setting the axiom **(GA.2)** is equivalent to the claim that $Dr_1(z,x)$ uniquely assembles to z.

M. DAVIS [1982], *Why Gödel didn't have Church's thesis*, ***Information and Control***, vol. 54, pp. 3–24.

R. GANDY [1980], *Church's thesis and principles for mechanisms*, ***The Kleene symposium*** (J. Barwise, H.J. Keisler, and K. Kunen, editors), North-Holland, pp. 123–148.

R. GANDY [1988], *The confluence of ideas in 1936*, ***The universal Turing machine—a half-century survey*** (R. Herken, editor), Oxford University Press, pp. 55–111.

K. GÖDEL [1933], *The present situation in the foundations of mathematics*, ***Collected works***, vol. III, Oxford University Press, pp. 45–53.

K. GÖDEL [1934], *On undecidable propositions of formal mathematical systems*, ***Collected works***, vol. I, Oxford University Press, pp. 356–371.

K. GÖDEL [1936], *Über die Länge von Beweisen*, ***Collected works***, vol. I, Oxford University Press, pp. 396–399.

K. GÖDEL [1946], *Remarks before the Princeton bicentennial conference on problems in mathematics*, ***Collected works***, vol. II, Oxford University Press, pp. 150–153.

K. GÖDEL [1972], *Some remarks on the undecidability results*, ***Collected works***, vol. II, Oxford University Press, pp. 305–306.

T. HERRON [1995], *An alternative definition of pushout diagrams and their use in characterizing K-graph machines*, Carnegie Mellon University.

D. HILBERT AND P. BERNAYS [1939], ***Grundlagen der Mathematik II***, Springer Verlag, Berlin.

S. C. KLEENE [1936], *General recursive functions of natural numbers*, ***Mathematische Annalen***, vol. 112, pp. 727–742, reprinted in (Davis 1965).

A. KOLMOGOROV AND V. USPENSKY [1963], *On the definition of an algorithm*, ***AMS Translations***, vol. 21, no. 2, pp. 217–245.

L. LAMPORT AND N. LYNCH [1990], *Distributed computing: Models and methods*, ***Handbook of theoretical computer science*** (J. van Leeuwen, editor), Elsevier Science Publisher B. V..

D. MUNDICI AND W. SIEG [1995], *Paper machines*, ***Philosophia Mathematica***, vol. 3, pp. 5–30.

N. DE PISAPIA [2000], *Gandy machines: an abstract model of parallel computation for Turing machines, the game of life, and artificial neural networks*, ***Master's thesis***, Carnegie Mellon Univeristy, Department of Philosophy, 75 pp.

E. POST [1947], *Recursive unsolvability of a problem of Thue*, ***The Journal of Symbolic Logic***, vol. 12, pp. 1–11.

J. SHEPHERDSON [1988], *Mechanisms for computing over arbitrary structures*, ***The universal Turing machine—a half-century survey*** (R. Herken, editor), Oxford University Press, pp. 581–601.

W. SIEG [1994], *Mechanical procedures and mathematical experience*, ***Mathematics and mind*** (A. George, editor), Oxford University Press, pp. 71–117.

W. SIEG [1996], *Aspects of mathematical experience*, ***Philosophy of mathematics today*** (E. Agazzi and G. Darvas, editors), Kluwer.

W. SIEG [1997], *Step by resursive step: Church's analysis of effective calculability*, ***The Bulletin of Symbolic Logic***, vol. 3, pp. 154–180.

W. SIEG [2000], *Calculations by man & machine: mathematical presentation*, Kluwer, to appear in: ***Proceedings of the 11th International Congress of Logic, Methodology and Philosophy of Science***.

W. SIEG AND J. BYRNES [1996], *K-graph machines: generalizing Turing's machines and arguments*, ***Gödel '96*** (P. Hajek, editor), Lecture Notes in Logic 6, Springer Verlag, pp. 98–119.

W. SIEG AND J. BYRNES [1999a], *Gödel, Turing, & K-graph machines*, ***Logic and the foundations of mathematics*** (A. Cantini, E. Casari, and P. Minari, editors), Synthese Library 280, Kluwer, (The paper was submitted for this volume already in January of 1997), pp. 57–66.

W. SIEG AND J. BYRNES [1999b], *An abstract model for parallel computations: Gandy's thesis*, ***The Monist***, vol. 82, no. 1, pp. 150–64.

H. T. SIEGELMANN [1998], ***Neural networks and analog computation—beyond the Turing limit***, Birkhäuser.

H. SINACEUR [2000], *Address at the Princeton University bicentennial conference on problems of mathematics (December 17–19, 1946), by Alfred Tarski*, ***The Bulletin of Symbolic Logic***, vol. 6, no. 1, pp. 1–44.

R. SOARE [1996], *Computability and recursion*, ***The Bulletin of Symbolic Logic***, vol. 2, no. 3, pp. 284–321.

G. TAMBURRINI [1987], *Reflections on mechanism*, ***Ph.D. thesis***, Columbia University, Department of Philosophy, New York.

G. TAMBURRINI [1997], *Mechanistic theories in cognitive science: The import of Turing's thesis*, ***Logic and scientific methods*** (M. L. Dalla Chiara, K. Doets, D. Mundici, and J. van Benthem, editors), Synthese Library 259, Kluwer, pp. 239–57.

A. TURING [1936], *On computable numbers with an application to the Entscheidungsproblem*, ***Proceedings of the London Mathematical Society***, 2, vol. 42, pp. 230–265.

A. TURING [1939], *Systems of logic based on ordinals*, ***Proceedings of the London Mathematical Society***, 2, vol. 45, pp. 161–228.

A. TURING [1950a], *Computing machinery and intelligence*, ***Mind***, vol. 59, pp. 433–460.

A. TURING [1950b], *The word problem in semi-groups with cancellation*, ***Annals of Mathematics***, vol. 52, pp. 491–505.

A. TURING [1953], *Solvable and unsolvable problems*, ***Science News***, vol. 31, pp. 7–23.

H. WANG [1974], ***From mathematics to philosophy***, Routledge & Kegan Paul, London.

DEPARTMENT OF PHILOSOPHY
CARNEGIE MELLON UNIVERSITY
PITTSBURGH, PENNSYLVANIA 15213, USA
E-mail: ws15+@andrew.cmu.edu