Calculations by Man & Machine

Mathematical presentation

Wilfried Sieg

April 5, 2000

Technical Report No. CMU-PHIL-105

Philosophy

Methodology

Logic

Carnegie Mellon

Pittsburgh, Pennsylvania 15213

Calculations by Man & Machine:
mathematical presentation*

Wilfried Sieg
Carnegie Mellon University
Pittsburgh, PA 15213

0. Introduction. These investigations aim to provide a characterization of computations by machines that is as general and convincing as that of computations by human computors given by Turing. The groundwork was laid by Gandy in his thought-provoking 1980 paper *Church's Thesis and Principles for Mechanisms* -- a rich and difficult, but sometimes unnecessarily (and maddeningly) complex paper. The structure of Turing's argument actually guided Gandy's analysis. If Gandy had been concerned with just sequential computations, his analysis would have joined Turing's perfectly with *exactly one* difference: physical restrictions for a machine, instead of sensory limitations of a human computor, would have motivated crucial boundedness and locality conditions. However, Gandy realized through conversations with J. C. Shepherdson that the analysis "must take parallel working into account".[1]

In a comprehensive survey article, published ten years after Gandy's paper, Leslie Lamport and Nancy Lynch argued that the theory of sequential computing "rests on fundamental concepts of computability that are independent of any particular computational model". They emphasized that, in contrast, the "fundamental formal concepts underlying distributed computing", if there were any, had not yet been developed. "Nevertheless", they wrote, "one can make some informal observations that seem to be important":

> 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. (p. 1166)

Gandy's analysis of parallel computations is conceptually convincing and provides a sharp mathematical form of the informal assumption(s) "underlying almost all models of concurrent systems". Gandy takes as the paradigmatic parallel computation the evolution of the Game of Life or of other cellular automata.

The definitional preliminaries in Gandy's paper are rather lengthy; Shepherdson wrote in 1988: "Although Gandy's principles 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." The "nature of the situation" is actually not as complex. The presentation can be simplified by choosing definitions appropriately and focusing sharply on the central informal ideas. Steps in clarifying and streamlining Gandy's mathematical development were taken in earlier work by John Byrnes and me.[2] In early 1999, I succeeded in simplifying further the mathematical apparatus, but my investigations took also a methodological turn that is articulated in (Sieg 2000). The conceptual analysis of calculations by man and machine given there, is complemented here by a systematic mathematical presentation.

Having learned Turing's lesson, that we have to take into account the computing agent or device (when analyzing a notion of computability), we have to ask "Parallel computations by whom or what?". Gandy focused on *discrete mechanical devices* possibly excluding other physical devices, for example, analogue machines. The only physical presuppositions about such devices are a lower bound on the size of atomic components and an upper bound on the speed of signal propagation.[3] Calculations proceed in discrete steps; thus such mechanical devices are "in a loose sense" digital computers. Finally, the behavior of these devices is uniquely determined, once a full description of their initial state is given, i.e., they are deterministic. After setting up in section 1 the basic mathematical frame and defining *Turing Computors*, I give in section 2 a straightforward, but re-oriented definition of *Gandy Machines.*

I view the definition of a Gandy machine as an abstract definition of the same character as that of a group or topological space, with many interestingly different models. The *only* axiomatic principle characterizing these machines is a restricted version of Gandy's principle of local causation. The proof of the central theorem[4], needed for the reduction of Gandy to Turing machines, is corrected and presented in section 3. It is the crucial step to obtain a suitable representation theorem: any process carried out by a model of the axioms, i.e., by a particular Gandy machine, can be simulated by a Turing machine. My concluding remarks in section 4 contain that formulation and some further observations.

1. Turing's computor. The crucial step, taken by Post and Turing in their analysis of calculability, was this: they moved away from directly characterizing calculable number theoretic functions (as proposed, for example, by Church using Gödel's equational calculus) and turned attention to the underlying mechanical operations on finite syntactic configurations; the story of this development is told in my (1994) and also, briefly, in (Sieg 2000). The computation models proposed by Turing and Post in 1936 were almost identical, but only Turing argued for the adequacy of his model; for Post it was a "working hypothesis". The real systematic confluence of Turing's and Post's work occurred in 1947, when Post gave a most elegant way of decribing Turing machines via his production systems (on the way to solving, negatively, the word-problem for semi-groups).[5] The configurations of a Turing machine are given by *instantaneous descriptions* (or ids) of the form $\alpha q_l s_k \beta$, where α and β are possibly empty strings of symbols in the machines's alphabet; more precisely, an id contains exactly one state symbol q_l, and to the right of it there must be at least one symbol s_k. Such ids express that the current tape content is $\alpha s_k \beta$, the machine is in state q_l, and it scans a square with symbol s_k. Quadruples $q_i s_k c_l q_m$ of the machine's program express that the machine, when observing symbol s_k in internal state q_i, is to carry out operation c_l and change its internal state to q_m. Such quadruples are incorporated into rules operating on ids; for example, if the operation c_l is the machine's command *print* 0, the corresponding rule is:

$$\alpha q_l s_k \beta \Rightarrow \alpha q_m 0 \beta.$$

Such rules can be formulated, obviously, for all the different operations of a Turing machine. One just has to append 0 to α (β) in case c_l is the operation *move to the left* (*right*) and α (β) is the empty string; this reflects the expansion of the finite, but potentially infinite tape by a blank square.[6]

Coming back to Turing's conceptual analysis: it attempts to arrive at the most basic, local operations on a bounded number of configurations. If one combines this with the Post presentation of Turing machines, restrictive conditions can be stated in great generality 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:

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

Discrete dynamical systems provide an elegant way of formulating these considerations mathematically. Let **D** be a class of states (ids or syntactic configurations), and let **F** be a function from **D** to **D** transforming a given state into the next one, i.e., describing the evolution of the system (the computation of the machine). Using already Gandy's framework, states are presented as non-empty hereditarily finite sets over an infinite set **U** of atoms. Such sets reflect states of physical devices just as mathematical structures represent states of nature. Thus, any isomorphic structure will do as well; one should notice that this reflection is done somewhat indirectly, as only the $\in$–relation is available.

A class **S** of states is called *structural*, if **S** is closed under $\in$-isomorphisms; if **D** is a class of states, I denote by $\mathbf{S}_\mathbf{D}$ the corresponding structural class. The lawlike connections between states are given by *structural operations* **G** from **S** to **S**. Structural operations satisfy the following condition:[7] for all permutations π on **U** and all $\mathbf{x} \in \mathbf{S}$, $\mathbf{G}(\mathbf{x}^{\pi})$ is $\in$-isomorphic over $\mathbf{x}^{\pi}$ to $\mathbf{G}(\mathbf{x})^{\pi}$; for any state **z**, $\mathbf{z}^{\pi}$ stands for $\pi^*(\mathbf{z})$, where π^* is the uniquely determined $\in$–isomorphism on the hereditarily finite sets extending π. "**x** is $\in$-isomorphic over **z** to **y**" means that the $\in$-isomorphism between **x** and **y** is the identity on the atoms in the transitive closure of **z**, i.e., in the support of **z** (briefly: Sup(**z**)). I will use the abbreviation "$\mathbf{x} \equiv_{\mathbf{z}} \mathbf{y}$"; if a special underlying permutation σ is to be pointed out, I write "$\mathbf{x} \equiv_{\mathbf{z}} \mathbf{y}$ via σ".

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 the computor can operate. Proper subtrees **y** of the $\in$-tree for **x** are called *parts for* **x**, briefly $\mathbf{y} <^* \mathbf{x}$, if they are specified as follows[8]: $\mathbf{y} \neq \mathbf{x}$ and **y** is a non-empty subset of

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

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 $\{\mathbf{r}\}^n$, 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 $\mathbf{z} \in \mathbf{C}$, such that **y** is a path through **z**. These last two definitions are not needed for the definition of a Turing computor; also the following definitions will be given in greater generality than needed for that purpose. (I just want to avoid duplicating matters for Gandy machines in the next section.)

The local operations are given by a structural operation **G** that works on certain parts **y** for **x**. Each **y** lies in one of a finite number of isomorphism classes (or stereotypes). So let **T** be a fixed, finite class of stereotypes: a part for **x** that is a member of a stereotype of **T** is called, naturally enough, a **T**-*part for* **x**. A **T**-part **y** for **x** is a *causal neighborhood for* **x** given by **T**, briefly $\mathbf{y} \in \mathrm{Cn}(\mathbf{x})$, if there is no **T**-part $\mathbf{y}^*$ for **x** such that **y** is $\in$-embeddable into $\mathbf{y}^*$. **G** operates on such causal neighborhoods. The values of **G**, however, are in general not exactly what is needed for assembling the next state. For that purpose, we introduce *determined regions* of a state **z** obtained from causal neighborhoods for **x**: $\mathbf{v} \in \mathrm{Dr}(\mathbf{z},\mathbf{x})$ if and only if $\mathbf{v} <^* \mathbf{z}$ and there is a $\mathbf{y} \in \mathrm{Cn}(\mathbf{x})$, such that $\mathbf{G}(\mathbf{y}) \equiv_{\mathbf{y}} \mathbf{v}$ and $\mathrm{Sup}(\mathbf{v}) \cap \mathrm{Sup}(\mathbf{x}) \subseteq \mathrm{Sup}(\mathbf{y})$. The last condition for Dr guarantees that new atoms in **G**(**y**) correspond to new atoms in **v**, and that the new atoms in **v** are new for **x**. If one requires **G** to satisfy similarly $\mathrm{Sup}(\mathbf{G}(\mathbf{y})) \cap \mathrm{Sup}(\mathbf{x}) \subseteq \mathrm{Sup}(\mathbf{y})$, then the condition "$\mathbf{G}(\mathbf{y}) \equiv_{\mathbf{y}} \mathbf{v}$" can be strengthened to "$\mathbf{G}(\mathbf{y}) \equiv_{\mathbf{x}} \mathbf{v}$". The new atoms are thus always taken from $\mathbf{U} \setminus \mathrm{Sup}(\mathbf{x})$.[9] One final definition: for given states **z** and **x** let A(**z**,**x**) stand for $\mathrm{Sup}(\mathbf{z}) \setminus \mathrm{Sup}(\mathbf{x})$. Note that the number of new atoms introduced by **G** is bounded, i.e., $|A(\mathbf{G}(\mathbf{y}), \mathrm{Sup}(\mathbf{x}))| < n$ for some natural number n (any $\mathbf{x} \in \mathbf{S}$ and any causal neighborhood **y** for **x**). Now we define, finally, a Turing computor, keeping the Post-description and the general boundedness and locality conditions firmly in mind.

Definition: $\mathbf{M} = <\mathbf{S}; \mathbf{T}, \mathbf{G}>$ 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 $\mathbf{x} \in \mathbf{S}$ there is a $\mathbf{z} \in \mathbf{S}$, such that

(LC.0) $(\exists! \, \mathbf{y}) \ \mathbf{y} \in \mathrm{Cn}(\mathbf{x})$

(LC.1) $(\exists! \, \mathbf{v} <^* \mathbf{z}) \ \mathbf{v} \equiv_{\mathbf{x}} \mathbf{G}(\mathbf{cn}(\mathbf{x}))$;

(GA.1) $\mathbf{z} = (\mathbf{x} \setminus \mathbf{cn}(\mathbf{x})) \cup \mathbf{dr}(\mathbf{z},\mathbf{x})$.

cn(**x**) and **dr**(**z**,**x**) denote the sole causal neighborhood of **x**, respectively the determined region of **z**. - It is easily seen that Turing machines are Turing computors under an appropriate set theoretic representation; I indicate one such representation in the Appendix. Notice that the next state **z** is determined uniquely up to $\in$–isomorphism. I denote it by **M**(**x**) and define what it means for a dynamical system to be (Turing) computable. (Recall that $\mathbf{S}_{\mathbf{D}}$ is the structural class corresponding to **D**.)

Definition. Let <**D**, **F**> be a discrete dynamical system; **F** is called (Turing) *computable* if and only if there is a Turing computor **M** on $\mathbf{S}_{\mathbf{D}}$, such that for each $\mathbf{x} \in \mathbf{D}$: $\mathbf{F}(\mathbf{x}) \equiv_{\mathbf{x}} \mathbf{M}(\mathbf{x})$.

2. Parallel computing. Can we generalize, in a suitable way, these considerations from sequential to parallel computations? Keeping in mind the considerations reviewed in the Introduction, we want to have a computational device that determines the evolution of dynamical systems in parallel. In analogy to the definition of Turing computability above, assuming that the next state **z** of a Gandy machine is also determined uniquely up to $\in$–isomorphism over **x**, we set tentatively:

Definition. Let <**D**, **F**> be a discrete dynamical system; **F** is called *computable in parallel* if and only if there is a Gandy machine **M** on $\mathbf{S}_D$, such that for each $x \in \mathbf{D}$: $\mathbf{F}(x) \equiv_x \mathbf{M}(x)$.

Thus, in contrast to (Gandy 1980) and (Sieg & Byrnes 1999B), Gandy machines are no longer viewed as discrete dynamical systems satisfying certain restrictive principles, but rather as restricted devices to *compute in parallel* (the evolution of) such systems up to $\in$–isomorphism over **x**. In addition to the stereotypes and structural operation working on causal neighborhoods, we have here a second set of stereotypes and a second structural operation: the latter allow us to "assemble" the determined regions.

Definition: $\mathbf{M} = \langle \mathbf{S}; \mathbf{T}_1, \mathbf{G}_1, \mathbf{T}_2, \mathbf{G}_2 \rangle$ is a *Gandy Machine on* **S**, where **S** is a structural class, $\mathbf{T}_i$ a finite set of stereotypes, $\mathbf{G}_i$ a structural operation on $\mathbf{T}_i$, if and only if, for every $x \in \mathbf{S}$ there is a $z \in \mathbf{S}$, such that

(LC.1) $(\forall\, y \in Cn_1(\mathbf{x}))\ (\exists!\, v \in Dr_1(\mathbf{z},\mathbf{x}))\ v \equiv_x \mathbf{G}_1(y)$;

(LC.2) $(\forall\, y \in Cn_2(\mathbf{x}))\ (\exists\, v \in Dr_2(\mathbf{z},\mathbf{x}))\ v \equiv_x \mathbf{G}_2(y)$;

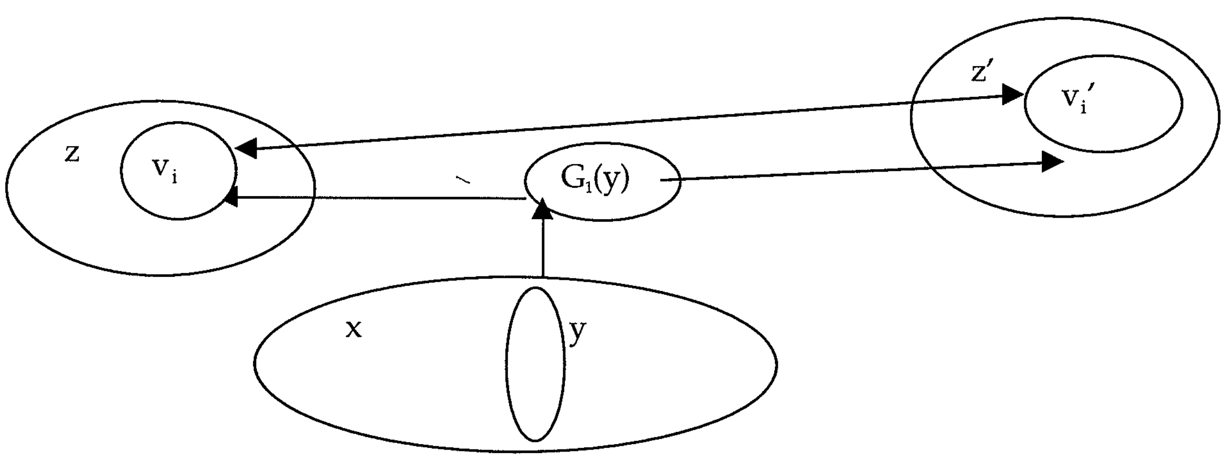
(GA.1) $(\forall\, C)\ [C \subseteq Dr_1(\mathbf{z},\mathbf{x}))\ \&\ \cap\{Sup(\mathbf{v}) \cap A(\mathbf{z},\mathbf{x}) \mid \mathbf{v} \in C\} \neq \emptyset \rightarrow$
$(\exists\, \mathbf{w} \in Dr_2(\mathbf{z},\mathbf{x}))\ (\forall\, \mathbf{v} \in C)\ \mathbf{v} <^* \mathbf{w}\ ]$;[10]

(GA.2) $\mathbf{z} = \cup\, Dr_1(\mathbf{z},\mathbf{x})$.

LC stands for *Local Causation*, whereas **GA** abbreviates *Global Assembly*. -- The central fact for Gandy machines (**z** is determined uniquely up to $\in$–isomorphism over **x**) follows easily from the next theorem[11]:

Theorem. Let **M** be $\langle \mathbf{S}; \mathbf{T}_1, \mathbf{G}_1, \mathbf{T}_2, \mathbf{G}_2 \rangle$ as above and $\mathbf{x} \in \mathbf{S}$; if there are **z** and **z'** in **S** satisfying principles (**LC.1-2**), (**GA.1**), and if $Dr_1(\mathbf{z},\mathbf{x})$ and $Dr_1(\mathbf{z}',\mathbf{x})$ cover **z** and **z'**, then $Dr_1(\mathbf{z},\mathbf{x}) \equiv_x Dr_1(\mathbf{z}',\mathbf{x})$.

In the following Dr_1, Dr_1', A, and A' will abbreviate $Dr_1(\mathbf{z},\mathbf{x})$, $Dr_1(\mathbf{z}',\mathbf{x})$, $A(\mathbf{z},\mathbf{x})$, and $A(\mathbf{z}',\mathbf{x})$ respectively. Note that Dr_1 and Dr_1' are finite. Using (**LC.1**) and (**LC.2**) one can observe that there is a natural number m and there are sequences $\mathbf{v}_i$ and $\mathbf{v}_i'$, $i<m$, such that $Dr_1 = \{\mathbf{v}_i \mid i<m\}$, $Dr_1' = \{\mathbf{v}_i' \mid i<m\}$, and $\mathbf{v}_i'$ is the unique part of **z'** with $\mathbf{v}_i \equiv_x \mathbf{v}_i'$ via permutations π_i (for all $i<m$). Here is a picture of the situation:



To establish the Theorem, we have to find a *single* permutation π that extends to an $\in$-isomorphism over **x** for all $\mathbf{v}_i$ and $\mathbf{v}_i'$ simultaneously. Such a π must obviously satisfy for all i<m:

(i) $\mathbf{v}_i \cong_x \mathbf{v}_i'$ via π

and, consequently,

(ii) $\pi[\mathrm{Sup}(\mathbf{v}_i)]=\mathrm{Sup}(\mathbf{v}_i')$;

As π is an $\in$-isomorphism over **x**, we have:

(iii) $\pi[A]=A'$.

Condition (ii) implies for all i<m and all $\mathbf{r}\in A$ the equivalence between $\mathbf{r}\in \mathrm{Sup}(\mathbf{v}_i)$ and $\mathbf{r}^\pi \in \mathrm{Sup}(\mathbf{v}_i')$. This can be also expressed by

(ii*) $\mu(\mathbf{r})=\mu'(\mathbf{r}^\pi)$, for all $\mathbf{r}\in A$,

where $\mu(\mathbf{r}) = \{i \mid \mathbf{r}\in \mathrm{Sup}(\mathbf{v}_i)\}$ and $\mu'(\mathbf{r}) = \{i \mid \mathbf{r}\in \mathrm{Sup}(\mathbf{v}_i')\}$; these are the *signatures* of **r** with respect to **z** and **z'**.

To obtain such a permutation, the considerations are roughly as follows: (i) if the $\mathbf{v}_i$ do not overlap, then the π_i will do; (ii) if there is overlap, then an equivalence relation $\approx$ ($\approx'$) on A (A') is defined by $\mathbf{r}_1 \approx \mathbf{r}_2$ iff $\mu(\mathbf{r}_1)=\mu(\mathbf{r}_2)$, and analogously for $\approx'$; (iii) then we prove that the "corresponding" equivalence classes $[\mathbf{r}]_\approx$ and $[\mathbf{s}]_{\approx'}$ (the signatures of their elements are identical) have the same cardinality. $[\mathbf{r}]_\approx$ can be characterized as $\cap\{\mathrm{Sup}(\mathbf{v}_i)\cap A \mid i\in\mu(\mathbf{r})\}$; similar for $[\mathbf{s}]_{\approx'}$. This characterization is clearly independent of the choice of representative by the very definition of the equivalence relation(s). With this in place, a global $\in$-isomorphism can be defined. These considerations are made precise through the proofs of the combinatorial lemma and two corollaries in the next section.

3. Global assembly. All considerations in this section are carried out under the assumptions of the Theorem: **M**=<**S**; $\mathbf{T}_1$, $\mathbf{G}_1$, $\mathbf{T}_2$, $\mathbf{G}_2$> is an arbitrary Gandy machine and $x\in \mathbf{S}$ an arbitrary state; we assume furthermore that **z** and **z'** are in **S**, the principles (**LC.1-2**) and (**GA.1**) are satisfied, and Dr_1 and Dr_1' cover **z** and **z'**. We want to show that $\mathrm{Dr}_1 \cong_x \mathrm{Dr}_1'$, knowing already that there are sequences $\mathbf{v}_i$ and $\mathbf{v}_i'$ of length m, such that $\mathrm{Dr}_1 = \{\mathbf{v}_i \mid i<m\}$, $\mathrm{Dr}_1' = \{\mathbf{v}_i' \mid i<m\}$, and $\mathbf{v}_i'$ is the unique part of **z'** with $\mathbf{v}_i \cong_x \mathbf{v}_i'$ via permutations π_i (for all i<m). I start out with the formulation of a key lemma concerning overlaps.

Overlap Lemma. Let $\mathbf{r}_0\in A$ and $\mu(\mathbf{r}_0)\neq\varnothing$; then there is a permutation ρ on **U** with $\mathbf{v}_i \cong_x \mathbf{v}_i'$ via ρ for all $i\in\mu(\mathbf{r}_0)$.
Proof. We have $\{\mathbf{v}_i \mid i\in\mu(\mathbf{r}_0)\} \subseteq \mathrm{Dr}_1$; as $\mathbf{r}_0$ is in A and in $\mathrm{Sup}(\mathbf{v}_i)$ for each $i\in\mu(\mathbf{r}_0)\neq\varnothing$, we have also that $\cap\{\mathrm{Sup}(\mathbf{v}_i)\cap A \mid i\in\mu(\mathbf{r}_0)\} \neq \varnothing$. The antecedent of (**GA.1**) is satisfied, and we conclude that there is a $\mathbf{w}\in \mathrm{Dr}_2$ such that $\mathbf{v}_i <^* \mathbf{w} <^* \mathbf{z}$, for all $i\in\mu(\mathbf{r}_0)$. Using (**LC.2**) we obtain a $\mathbf{w}'\in \mathrm{Dr}_2'$ with $\mathbf{w} \cong_x \mathbf{w}'$. This $\in$-isomorphism over **x** is induced by a permutation ρ and yields for all $i\in\mu(\mathbf{r}_0)$

$$\mathbf{v}_i^\rho <^* \mathbf{w}^\rho = \mathbf{w}' <^* \mathbf{z}'.$$

So we have, $\mathbf{v}_i \cong_x \mathbf{v}_i^\rho$ and $\mathbf{v}_i^\rho <^* \mathbf{z}'$, thus – using (**LC.2**) - $\mathbf{v}_i^\rho = \mathbf{v}_i'$; that holds for all $i\in\mu(\mathbf{r}_0)$. **Q.E.D.**

Note that the condition $\mu(\mathbf{r})\neq\varnothing$ is satisfied in our considerations for any $\mathbf{r}\in A$, as Dr_1 is a cover of **z**; so we have for any such **r** an appropiate *overlap permutation* ρ^r *for* **r**. The crucial combinatorial lemma we have to establish is this:

Combinatorial Lemma. For $\mathbf{r}_0\in A$: $|\{\mathbf{r}\in A \mid \mu(\mathbf{r}_0)\subseteq\mu(\mathbf{r})\}| = |\{\mathbf{s}\in A' \mid \mu(\mathbf{r}_0)\subseteq\mu'(\mathbf{s})\}|$.
Proof. Consider $\mathbf{r}_0\in A$. I establish first the claim

$$\rho[\{\mathbf{r}\in A \mid \mu(\mathbf{r}_0) \subseteq \mu(\mathbf{r})\}] \subseteq \{\mathbf{s}\in A' \mid \mu(\mathbf{r}_0) \subseteq \mu'(\mathbf{s})\},$$

where ρ is an overlap permutation for $\mathbf{r}_0$. The claim follows easily from

$$\mathbf{r}\in A \;\&\; \mu(\mathbf{r}_0) \subseteq \mu(\mathbf{r}) \rightarrow \mu(\mathbf{r}_0) \subseteq \mu'(\mathbf{r}^\rho),$$

by observing that $\mathbf{r}^\rho$ is in A'. Assume, to establish this conditional indirectly, for arbitrary $\mathbf{r}\in A$ that $\mu(\mathbf{r}_0)\subseteq\mu(\mathbf{r})$ and $\neg(\mu(\mathbf{r}_0)\subseteq\mu'(\mathbf{r}^\rho))$. The first assumption implies that $\mathbf{r}\in \mathrm{Sup}(\mathbf{v}_i)$ for all $i\in\mu(\mathbf{r}_0)$, and the construction of ρ yields then:

(♥) $r^\rho \in Sup(v_i')$ for all $i \in \mu(r_0)$.

The second assumption implies that there is a k in $\mu(r_0) \setminus \mu'(r^\rho)$. Obviously, $k \in \mu(r_0)$ and $k \notin \mu'(r^\rho)$. The first conjunct $k \in \mu(r_0)$ and (♥) imply that $r^\rho \in Sup(v_k')$; as the second conjunct $k \notin \mu'(r^\rho)$ means that $r^\rho \notin Sup(v_k')$, we have obtained a contradiction.

Now I'll show that $\rho[\{r \in A \mid \mu(r_0) \subseteq \mu(r)\}]$ cannot be a *proper* subset of $\{s \in A' \mid \mu(r_0) \subseteq \mu'(s)\}$. Assume, to obtain a contradiction, that it is; then there is $s^* \in A'$ that satisfies $\mu(r_0) \subseteq \mu'(s^*)$ and is not a member of $\rho[\{r \in A \mid \mu(r_0) \subseteq \mu(r)\}]$. As $\mu(r_0) \subseteq \mu'(s^*)$, s^* is in $Sup(v_i')$ for all $i \in \mu(r_0)$; the analogous fact holds for all $r \in A$ satisfying $\mu(r_0) \subseteq \mu(r)$, i.e., all such **r** must be in $Sup(v_i)$ for all $i \in \mu(r_0)$. As $v_i \cong_x v_i'$ via ρ for all $i \in \mu(r_0)$, s^* must be obtained as a ρ-image of some r^* in Sup(x) or in A (and, in the latter case, violating $\mu(r_0) \subseteq \mu(r^*)$). However, in either case we have a contradiction. The assertion of the Lemma is now immediate. **Q.E.D.**

Next I establish two consequences of the Combinatorial Lemma, the second of which is basic for the definition of the global isomorphism π.

Corollary 1. For any $I \subseteq \{0, 1, \dots, m\text{-}1\}$ with $I \subseteq \mu(r_0)$ for some r_0 in A,

$$|\{r \in A \mid I \subseteq \mu(r)\}| = |\{s \in A' \mid I \subseteq \mu'(s)\}| .$$

Proof. Consider an arbitrary $I \subseteq \mu(r_0)$ for some r_0 in A. If $I = \mu(r_0)$, then the claim follows directly from the Combinatorial Lemma. If $I \subset \mu(r_0)$, let $r^0, \dots, r^{k-1}$ be elements **r** of A with $I \subset \mu(r)$ and require that $\mu(r^j) \neq \mu(r^{j'})$, for all j, j'<k and j≠j', and for every $r \in A$ with $I \subset \mu(r)$ there is a unique j<k with $\mu(r) = \mu(r^j)$. The Combinatorial Lemma implies, for all j<k,

$$|\{r \in A \mid \mu(r^j) \subseteq \mu(r)\}| = |\{s \in A' \mid \mu(r^j) \subseteq \mu'(s)\}| .$$

Now it is easy to verify the claim of Corollary 1:

$|\{r \in A \mid I \subseteq \mu(r)\}| =$
$|\{r \in A \mid (\exists j<k)\ \mu(r^j) \subseteq \mu(r)\}| =$
$|\{s \in A' \mid (\exists j<k)\ \mu(r^j) \subseteq \mu'(s)\}| =$
$|\{s \in A' \mid I \subseteq \mu'(s)\}|$.

This completes the proof of Corollary 1. **Q.E.D.**

The second important consequence of the Combinatorial Lemma can be obtained now by an inductive argument.

Corollary 2. For any $I \subseteq \{0, 1, \dots, m\text{-}1\}$ with $I \subseteq \mu(r_0)$ for some r_0 in A

$$|\{r \in A \mid I = \mu(r)\}| = |\{s \in A' \mid I = \mu'(s)\}| .$$

Proof (by downward induction on $|I|$). Abbreviating $|\{r \in A \mid I = \mu(r)\}|$ by ν_I and $|\{s \in A' \mid I = \mu'(s)\}|$ by ν'_I, the argument is as follows:

Base case ($|I|=m$): In this case there are no proper extensions I* of I, and we have

$\nu_I = |\{r \in A \mid I = \mu(r)\}|$
$= |\{r \in A \mid I \subseteq \mu(r)\}|$, as there is no proper extension of I,
$= |\{s \in A' \mid I \subseteq \mu'(s)\}|$, by Corollary 1,
$= |\{s \in A' \mid I = \mu'(s)\}|$, again, as there is no proper extension,
$= \nu'_I$

Induction step ($|I|<m$): Assume that the claim holds for all I* with $n+1 \leq |I^*| \leq m$ and show that it holds for I with $|I|=n$. Using the induction hypothesis we have, summing up over all proper extensions I* of I:

$$(\clubsuit)\ \Sigma_{I^*} \nu_{I^*} = \Sigma_{I^*} \nu'_{I^*} .$$

Now we argue as before:

$\nu_I = |\{r \in A \mid I = \mu(r)\}|$
$= |\{r \in A \mid I \subseteq \mu(r)\}| - \Sigma_{I^*} \nu_{I^*}$
$= |\{s \in A' \mid I \subseteq \mu'(s)\}| - \Sigma_{I^*} \nu'_{I^*}$, by Corollary 1 and (♣),
$= |\{s \in A' \mid I = \mu'(s)\}|$
$= \nu'_I$

This completes the proof of Corollary 2. **Q.E.D.**

Finally, we can define an appropriate global permutation π. Given an atom $\mathbf{r} \in A$, there are only finitely many different overlap permutations, i.e., different on $\cup\{Sup(\mathbf{v}_i) \mid i \in \mu(\mathbf{r})\}$. Let us select one of them and call it ρ^r. That permutation can be restricted to $[\mathbf{r}]_{\approx} = \cap\{Sup(\mathbf{v}_i) \cap A \mid i \in \mu(\mathbf{r})\}$; let ρ^* denote this restriction. Because of Corollary 2, ρ^* is a bijection between $[\mathbf{r}]_{\approx}$ and $[\rho^*(\mathbf{r})]_{\approx'}$. - The desired global permutation is now defined as follows for any atom $\mathbf{r} \in \cup\{Sup(\mathbf{v}_i) \mid i<m\}$:

$$\pi(\mathbf{r}) = \begin{cases} \rho^*(\mathbf{r}) & \text{if } \mathbf{r} \in \cap\{ Sup(\mathbf{v}_i) \cap A \mid i \in \mu(\mathbf{r}) \} \\ \mathbf{r} & \text{otherwise} \end{cases}$$

π is a well-defined bijection with $\pi[A]=A'$ and $\mu(\mathbf{r})=\mu'(\mathbf{r}^{\pi})$. It remains to establish the ***Claim***: For all $i<m$, $\mathbf{v}_i \cong_x \mathbf{v}_i'$ via π. For the ***Proof*** consider an arbitrary $i<m$. By the basic set-up of our considerations, we have $\pi_i(\mathbf{v}_i)=\mathbf{v}_i'$. If $\mathbf{v}_i$ does not contain in its support an element of A, then π and π_i coincide; if $\mathbf{v}_I$'s support contains an element of A that is possibly even in an overlap, the argument proceeds as follows. Notice that *all* elements of $[\mathbf{r}]_{\approx}$ are in $Sup(\mathbf{v}_i)$ as soon as one $\mathbf{r} \in A$ is in $Sup(\mathbf{v}_i)$. Taking this fact into account and using the pruning operation[12] $\uparrow$ we have, by the definition of π and of $\mathbf{v}_i \uparrow [\mathbf{r}]_{\approx}$: $\pi(\mathbf{v}_i \uparrow [\mathbf{r}]_{\approx}) = \rho^*(\mathbf{v}_i \uparrow [\mathbf{r}]_{\approx})$. The definition of ρ^* and the fact that $\rho^r(\mathbf{v}_i)=\mathbf{v}_i'$ allow us to infer that $\rho^*(\mathbf{v}_i \uparrow [\mathbf{r}]_{\approx}) = \mathbf{v}_i' \uparrow [\rho^*(\mathbf{r})]_{\approx'}$. As $\mu'(\rho^*(\mathbf{r}))=\mu'(\pi_i(\mathbf{r}))$ $[=\mu(\mathbf{r})]$ we can extend this sequence of identities by $\mathbf{v}_i' \uparrow [\rho^*(\mathbf{r})]_{\approx'} = \mathbf{v}_i' \uparrow [\pi_i(\mathbf{r})]_{\approx'}$. Consequently, as $\pi_i(\mathbf{v}_i)=\mathbf{v}_i'$, we have $\mathbf{v}_i' \uparrow [\pi_i(\mathbf{r})]_{\approx'} = \pi_i(\mathbf{v}_i \uparrow [\mathbf{r}]_{\approx})$.

These considerations hold for all $\mathbf{r} \in Sup(\mathbf{v}_i) \cap A$; we can conclude $\pi(\mathbf{v}_i)=\pi_i(\mathbf{v}_i)$ and, with $\pi_i(\mathbf{v}_i)=\mathbf{v}_i'$, we have $\pi(\mathbf{v}_i)= \mathbf{v}_i'$. **Q.E.D.** (of claim)

This concludes, finally, the argument for the Theorem that was formulated already in section 2.

4. Concluding remarks. If the operation **F** of a dynamical system is computable in parallel by a Gandy machine **M**, then it is also Turing computable. It follows from the Theorem and (**GA.2**) that **M**(**x**) is uniquely determined, up to isomorphism, for any **M** and any **M**-state **x**: we are dealing only with (finitely many) finite objects, and the axiomatic conditions for a Gandy machine are decidable. Thus, a search will allow us to find **M**(**x**). This fact is best understood, it seems to me, as a representation theorem in the given axiomatic setting.

Let me reiterate, what I emphasized in (Sieg 2000). It may be that Gandy machines can be simplified further, for example, by a graph theoretic presentation or a category theoretic description.[13] 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 nets.[14] 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 (1999).

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 be crucial?". These are important and most appealing questions.

APPENDIX

In this appendix I sketch a set theoretic presentation of a Turing machine as a Turing computor, but also – even more briefly – that of the Game of Life as a Gandy machine. Consider a Turing machine with symbols $s_0, \ldots, s_k$ and internal states $q_0, \ldots, q_l$; its program is given as a finite list of quadruples of the form $q_i s_j c_k q_l$, expressing that the machine is going to perform action c_k and change into internal state q_l, when scanning symbol s_j in state q_i. The tape is identified with a set of overlapping pairs

$$\mathbf{Tp} := \{ \langle b,b\rangle, \langle b,c\rangle, \ldots, \langle d,e\rangle, \langle e,e\rangle \}$$

where b, c, d, e are distinct atoms; c is the leftmost square of the tape (with a possibly non-blank symbol on it), d its rightmost one. The symbols and states are represented by $\underline{s}_j := \{r\}^{(j+1)}$, $0 \le j \le k$; the internal states are given by $\underline{q}_j := \{r\}^{(k+1)+(j+1)}$, $0 \le j \le l$. The tape content is given by

$$\mathbf{Ct} := \{ \langle \underline{s}_{j0},c\rangle, \ldots, \langle \underline{s}_{jr},d\rangle \}$$

and, finally, the id is represented as the union of **Tp**, **Ct**, and $\{ \langle \underline{q}_i, c\rangle \}$. So the structural set **S** of states is obtained as the set of all ids closed under $\in$–isomorphisms. Stereotypes (for each program line given by $q_i s_j$) consist of parts like

$$\{ \langle \underline{q}_i,r\rangle, \langle \underline{s}_j,r\rangle, \langle t,r\rangle, \langle r,u\rangle \};$$

these are the causal neighborhoods on which **G** is to operate. Consider the program line $q_i s_j s_k q_l$ (print s_k); applied to the above causal neighborhood **G** is to yield

$$\{ \langle \underline{q}_l,r\rangle, \langle \underline{s}_k,r\rangle, \langle t,r\rangle, \langle r,u\rangle \}.$$

For the program line $q_i s_j R q_l$ (move Right) two cases have to be distinguished. In the first case, when r is not the rightmost square, **G** yields

$$\{ \langle \underline{q}_l,u\rangle, \langle \underline{s}_j,r\rangle, \langle t,r\rangle, \langle r,u\rangle \};$$

in the second case, when r is the rightmost square, **G** yields

$$\{ \langle \underline{q}_l,*\rangle, \langle \underline{s}_j,r\rangle, \langle \underline{s}_0,*\rangle, \langle t,r\rangle, \langle r,*\rangle, \langle *,u\rangle \};$$

where $*$ is a new atom. The program line $q_i s_j L q_l$ (move Left) is treated similarly. - It is easy to verify that a Turing machine presented in this way is a Turing Computor.

Gandy considered playing Conway's Game of Life as a paradigmatic case of parallel computing. It is being played on subsets of the plane, more precisely, subsets that are constituted by finitely many connected squares. For reasons that will be obvious in a moment, the squares are also called *internal cells;* they can be in two states, *dead* or *alive*. In my presentation the internal cells are surrounded by one layer of *border cells,* the latter in turn by an additional layer of *virtual cells.* Border and virtual cells are dead by convention. Internal cells and border cells are jointly called *real.* The layering ensures that each real cell is surrounded by a full set of eight neighboring cells. For real cells the game is played according to the rules:

(1) living cells with 0 or 1 (living) neighbor die (from isolation);
(2) living cells with 4 or more (living) neighbors die (from overcrowding);
(3) dead cells with exactly 3 (living) neighbors become alive.

In all other cases the cell's state is unchanged.

A real cell a with neighbors $a_1, \ldots, a_8$ and state s(a) is given by

$$\{ a, s(a), \langle a_1, \ldots, a_8 \rangle \}.$$

(The neighbors are given in "canonical" order starting with the square in the leftmost top corner and proceeding clockwise; s(a) is {a} in case a is alive, otherwise {{a}}.) The $\mathbf{T}_1$-causal neighborhoods of real cells are of the form

$$\{ \{ a, s(a), \langle a_1, \ldots, a_8 \rangle\}, \{a_1, s(a_1)\}, \ldots, \{ a_8, s(a_8)\} \}.$$

It is obvious, how to define the structural operation G_1 on the causal neighborhoods of internal cells; the case of border cells requires attention. There is a big number of stereotypes that have to be treated, so I will discuss only one simple case that should, nevertheless, bring out the principled considerations. In the following diagram we start out with the cells that have letters assigned to them; the diagram should be thought of extending at the left and at the bottom. The v's indicate virtual cells, the b's border cells, the {a}'s internal cells that are alive, and the $*$'s new atoms that are added in the next step of the computation. Let's see how that comes about.

$*_0$	$*_1$	$*_2$	$*_3$	$*_4$	$*_5$	$*_6$	$*_7$		
v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	v_{10}
$\{a_0\}$	$\{a_1\}$	$\{a_2\}$	$\{a_3\}$	$\{a_4\}$	$\{a_5\}$	$\{a_6\}$	$\{a_7\}$	b_9	v_{11}

Consider the darkly shaded square b_3 with its neighbors, i.e., its presentation

$$\{ b_3, \{\{b_3\}\}, <v_2, \dots , b_2> \};$$

applying G_1 to its causal neighborhood yields

$$\{ \{ b_3, \{b_3\}, < v_2, \dots , b_2> \}, \{v_3, \{\{v_3\}\}, < *_2, *_3, *_4, v_4, b_4, b_3, b_2, v_2 >\} \},$$

where $*_2$, $*_3$, and $*_4$ are new atoms (and v_3 has been turned from a virtual cell into a real one, namely a border cell). Here the second set of stereotypes and the second structural operation come in to ensure that the new squares introduced by applying G_1 to "adjacent" border cells (whose neighborhoods overlap with the neighborhood of b_3) are properly identified in the next state. Consider as the appropriate $\mathbf{T}_2$-causal neighborhood the set consisting of the $\mathbf{T}_1$-causal neighborhoods of b_2, b_3, and b_4; G_2 applied to it yields the set with presentations of the cells v_2, v_3, and v_4.

---

* During the International Congress I had fruitful and engaging discussions with Yuri Gurevich, mostly, on the conceptual issues underlying this paper. – A comparative study of Gurevich's ASMs, but also of standard models of parallel computations (as in Brassard and Bratley's book) is the topic of a third paper on "Calculations by man & machine" with the subtitle "Comparisons and applications".

NOTES

1 (Gandy 1980), p. 125.

2 (Sieg 1990), (Byrnes 1993), and (Sieg & Byrnes 1999B).

3 That is emphasized on p. 126, but also on pp. 135-6 of (Gandy 1980). – How the purely physical considerations lead to boundedness and locality conditions is discussed at great length in (Mundici & Sieg).

4 That is Lemma 5.5 on p. 144 of (Gandy 1980).

5 Post's way of looking at Turing machines underlies also the presentation of (Davis 1958); for a more detailed discussion the reader is referred to that classical text.

6 Turing recognized the significance of Post's presentation for achieving mathematical results, but also for the conceptual analysis of calculability: as to the former, Turing extended in his (1950) Post's and Markov's result concerning the unsolvability of the word problem for semi-groups to semi-groups with cancellation; as to the latter, the reader should consult the semi-popular and wonderfully informative presentation of *Solvable and Unsolvable Problems* (1953).

7 For motivation of this particular condition see p. 154 of (Sieg & Byrnes 1999B) and p. 128 of (Gandy 1980).

8 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 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 $(\exists s \subseteq Tc(y))\ u = y \uparrow s$.

9 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.

10 In Gandy's set-up the finiteness of the **C** has to be explicitly required; here it is a trivial consequence of the finiteness of Dr_1. Furthermore, principle (**GA.1**) implies a fixed upper bound on the number of determined

regions that have new atoms in common, as there is clearly an upper bound on the number of distinct parts of elements Dr_2(**z**,**x**).

[11] In (Gandy 1980) this uniqueness up to ∈–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**; for details see (Sieg & Byrnes 1999B), p. 157. In my setting, axiom (**GA.2**) is equivalent to the claim that Dr_1(**z**,**x**) uniquely assembles to **z**.

[12] The pruning operation applies to an element **x** of HF and a subset **Y** of its support: **x**↑**Y** is the subtree of **x** that is built up exclusively from atoms in **Y**. The ∈–recursive definition is: (**x**∩**Y**) ∪ [{**y**↑(**Y**∩Tc(**y**)) | **y**∈**x** } \ {∅}]. Cf. (Sieg and Byrnes 1999b), pp. 155-6.

[13] 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).

[14] 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.

REFERENCES

G. Brassard and Bratley, P.: 1996 *Fundamentals of Algorithmics*; Prentice Hall.

Byrnes, J.: 1993 *Notions of Computation*; M.S. Thesis, Carnegie Mellon University, Pittsburgh.

Byrnes, J. and Sieg, W.: 1996 'A graphical presentation of Gandy's parallel machines' (Abstract), Bulletin of Symbolic Logic 2, 452-3

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

De Pisapia, N.: 2000 *Gandy Machines: an abstract model of parallel computation for Turing Machines, the Game of Life, and Artificial Neural Networks*; M.S. Thesis, Carnegie Mellon University, Pittsburgh.

Gandy, R.: 1980 'Church's Thesis and principles for mechanisms'; in: *The Kleene Symposium* (edited by J. Barwise, H.J. Keisler and K. Kunen, North-Holland, 123-148.

Gurevich, Y.: 1999 'Sequential ASM Thesis'; MSR-TR-99-09; Microsoft Research, Redmond.

Herron, T.: 1995 'An alternative definition of pushout diagrams and their use in characterizing K-graph machines'; Carnegie Mellon University.

Kolmogorov, A. and Uspensky, V.: 1963 'On the definition of an algorithm'; AMS Translations 21, 2, 217-45.

Lamport, L. and Lynch, N.: 1990 'Distributed Computing: Models and Methods'; in: *Handbook of Theoretical Computer Science* (J. van Leeuwen, ed.), Elsevier Science Publisher B.V.

Mundici, D. and Sieg, W.: 1995 'Paper machines'; Philosophia Mathematica 3, 5-30.

Post, E.: 1936 'Finite combinatory processes. Formulation 1'; Journal of Symbolic Logic1, 103-105.
-- .: 1947 'Recursive unsolvability of a problem of Thue'; J. Symbolic Logic 12, 1-11.

Shepherdson, J.C.: 1988 'Mechanisms for computing over arbitrary structures', in: *The Universal Turing Machine – A Half-Century Survey* (R. Herken, ed.), Oxford University Press, 581-601.

Sieg, W.: 2000 'Calculations by man & machines: conceptual analysis'; Technical Report 104, Philosophy, Methodology, Logic; to appear in: *Reflections*, Proc. Symposium in honor of S. Feferman's 70th Birthday.

Sieg, W. and Byrnes, J.: 1999A 'Gödel, Turing, & K-graph machines'; in: *Logic and Foundations of Mathematics* (A. Cantini, E. Casari, P. Minari, eds.), Synthese Library 280, Kluwer, 57-66.
-- . : 1999B 'An abstract model for parallel computations: Gandy's Thesis'; The Monist, 82 (1), 150-64.

Siegelmann, H.T.: 1999 *Neural Networks and Analog Computation – Beyond the Turing Limit*; Birkhäuser, Boston, Basel, Berlin.

Tamburrini, G.: 1987 *Reflections on Mechanism*; Ph.D. Thesis, Department of Philosophy, Columbia University, New York.

Turing, A.: 1936 'On computable numbers, with an application to the Entscheidungsproblem'; Proc. London Math. Soc., series 2, 42, 230-265.
-- .: 1950 'The word problem in semi-groups with cancellation'; Ann. of Math. 52, 491-505.
-- .: 1953 'Solvable and unsolvable problems'; Science News 31, 7-23.