The difference between Clocks and Turing Machines[1].

Giuseppe Longo

CNRS and Dépt. de Mathématiques et Informatique
Ecole Normale Supérieure, Paris

Content:

1. Clocks and Logic.
2. Turing Machines and the distinction between Hardware and Software
3. Simulating Turing Machines by Clocks.
4. Finite representations in Physics and in Metamathematics.
5. Mathematics as an open system.
6. Interactive, asynchronous and distributed computing.
7. Plasticity and the wet brain.
8. Implicit Mathematics
APPENDIX: More on the proofs of unprovable propositions.

1. **Clocks and Logic.**

During several centuries the prevailing metaphor for human brain was a "clock" This metaphor became precise with Descartes. For Descartes, there is a mechanical body and brain, a << statue d'automate>>, similar to the fantastic automatic devices of his time and ruled by cog-wheels, gears and pulleys. Separated from it, but ruling it, is the << res cogitans >>. The physical connecting point of this dualism was provided by the pineal gland; the philosophical compatibility, in my opinion, was given by the fact that the res cogitans too is governed by rules, logical/deductive ones. << Non evident knowledge may be known with certainty, provided that it is deduced from true and known principles, by a continually and never interrupted movement of thought which has a clear intuition of each individual step>>. Knowledge is a sequence or << a chain >> where << the last ring ... is connected to the first>> [Descartes,1619-1664; rule III]. In view of their role in the mathematical work of Descartes, it may be sound to consider these remarks as the beginning of Proof Theory: mathematics is no longer (or not only) the revelation or inspection of an existing reality, where "proofs" (mostly informal and incomplete, often wrong) were only meant to display

[1] Conference on "**Models of Cognition and Complexity Theory**", invited lecture, Rome, November 1994. Proceedings in **La Nuova Critica**, 29 (1), pp. 31-42, 1995.

"truth", but it is based on the manipulation of algebraic symbols and stepwise deductions from evident knowledge. Descartes' Analytic Geometry, an algebraic approach to Geometry, brought the entire realm of mathematics under the control of formal deductions and algebraic computations, as distinct from the geometrical observation. In Algebra and, thus, in Algebraic or Analytic Geometry, proofs are sequences of equations, formally manipulated, independently of their (geometric) meaning. Three centuries later, Proof Theory will be the rigorous (mathematical) description of these deductions.

Consider now that mathematics is, for Descartes, the paradigm, the highest level, of human thinking. This is why, in spite of Descartes' dualism, there is a great unity in his understanding of mind, from a modern perspective: both the res cogitans and its physical support obey "formal" or "mechanical" rules. Their description is derived from the concrete experience of the clocks and automata of the time as well as from Descartes' early steps towards modern metamathematical reflections and mathematical rigor, largely related to the nature of Analytic Geometry.

In the rationalist tradition, the connection between mechanical thinking and physical brain went much further. In Diderot and D'Alembert, clocks are more than a metaphor for brain (and mind): they provide a precise scientific reductionist project, in a monist perspective. The philosopher and the harpsichord have the same degree of materiality: the same mechanical rules govern human thinking and the musical instrument. The logical syllogism works like a machine and the machine works like a syllogism [Buffat, 1982]. The encyclopedists' radical mechanicalism aims at a *complete*, mechanical theory of the world: perfect, "encyclopedic" knowledge is possible by the understanding of all the rules of deduction and all clocklike mechanisms of matter (<< les phénomènes sont nécessaires >> in mathematics and in physics). Vaucanson's fantastic "Canard", which could eat, drink and walk, his "Music Players" were the first steps towards the concrete realization of this project.

The reduction was not proposed in a shallow way: these authors hinted at a mechanical understanding of human functions and mind in the full awareness of the limits of the "clocks" they knew. They essentially meant to underline the need for a reduction of the complexity of human mental activity to scientifically describable facts, while aware that a lot more chemistry and physics had yet to be understood. Moreover, << les machines sensibles >> had to be based on a great psycho-somatic unity: for Diderot, there is no intelligence without joy and suffering, without sexuality and feelings [Diderot, 1973]. But yet they thought that, in the end, the mechanics of all of this could be somehow fully described by a material machine, whose physical structure contained all elements of this reduction. The difference between brain and clocks was only quantitative, a pure matter of complexity, we would say today.

This philosophical perspective cannot be distinguished from the progress in mathematics, in the XVIII century, and from the growth of mathematical rigor. Leibniz' Mathesis Universalis, or the representation of the world in mathematical terms, is in the background. The philosopher is a << machine à reflexions >> exactly because he can inspect the world by the rules of mathematics and these rules can be implemented in clocks. Shortly later, with Laplace, the highest point of this view is reached: the Universe is similar to a perfect immense

“clock”, ruled by Newton's mathematically describable laws.

Laplace's philosophy largely dominated the XIX century's science, when Boole, Babbage and Power worked. The influence is clear: the complete descriptions by mathematical laws may cover the world and, in particular, human reasoning. Boole's "Laws of Thought" provide the first algebraic counterpart of Descartes' still informal metamathematics: mathematical thinking is being mathematized. Thus, better implementations in machines can be made of mathematical, indeed human, reasoning. Babbage and Power enriched clocks and Pascal's arithmetic machine, by multiple possible choices and combinations in the pieces of hardware: besides arithmetic operations, their machines could perform differentiation by a technique that we could call today a "finite elements method". In Power's machines, some parts of the hardware were as changeable as holes in strips of paper... The connection between the progress in Logic and machines was still informal and depended more on a common cultural frame, as almost a century is needed before boolean logic could be implemented in electronic circuits. The background was given by the major progress of mathematics and mathematical rigor in the XIX century. Thus, once more, the mechanics of thinking machines followed or grew together with the understanding of (meta)mathematics. A dramatic change will be induced by further advances in Logic at the turn of the century.

2. **Turing Machines and the distinction between Hardware and Software**

Frege is commonly considered as the father of modern Mathematical Logic. By his work, and that of Dedekind, Peano and Hilbert, metamathematics was finally described essentially as we know it today. At the base of this mathematization of mathematical deduction, there is the key distinction between language and metalanguage. Mathematics, the object of study, is carried on in an object language, rigorously investigated in a metalanguage, by the newly specified tools of Proof Theory. The aim is similar to Laplace's, at a "higher" level: according to Laplace, by the language of mathematics (the partial differential equations of Analysis) one can completely describe and predict the Universe. Similarly, by the formal tools of metamathematics one should completely describe and decide mathematics (Hilbert's conjectures of completeness, decidability and provable consistency for formalized Analysis).

The mathematics of metamathematics though has a peculiarity, stressed by Hilbert: it is finitistic, as it must use finitely represented languages and finite procedures, such as deductions. The point then was to make precise what this finitistic approach could mean and, in particular, the exact extent of finite procedures. Within a Hilbertian perspective, in 1930-36, Gödel, Herbrand, Kleene, Church and Curry proposed a precise mathematical notion of "effective or algorithmic function" (indeed, several ones, all proved equivalent by Kleene and Turing, in 1936). In a sense, they formally described, as an algorithmic procedure, Descartes's early hints towards the stepwise process of human deduction; as a result, they obtained a mathematically sound theory of computable functions. All their different theories were original axiomatic approaches to brand new and deep mathematics, based primarily on functions, more than on sets, but still in the frame of familiar mathematical languages and

tools (Herbrand-Gödel recursive functions, Church's lambda-calculus, Curry's Combinatory Logic ...). Turing, in 1936, had an even more revolutionary idea: the invention of an abstract, mathematical, machine (TM), only vaguely inspired by previous mechanical devices, but absolutely unrelated, a priori, to mathematical theories of functions, sets or whatever. The core of his idea was the (abstract) distinction between "hardware" and "software": on one side the unchangeable material basis (a "read/write" head, moving on a tape), on the other, the changeable sets of instructions (the programs, as finite lists of instructions: move right/left, write/erase a symbol on the tape...). One can notice here the same distinction, as in metamathematics, between metalanguage and object language, as well as Tarski's precise partition between syntax (the stepwise computations of a formal TM) and semantics (the class of computable functions, in extenso). The equivalence of the Turing computable functions with the mathematical approaches proposed by the others, listed above, is still now a surprising, deep idea and a technically difficult theorem (though very boring). As a result, a finite machine, though abstract, could compute infinitely many functions, by the invention of "software". Moreover, each of these formalisms possesses a "universal" function or machine: as a matter of fact, any finitely describable algorithm or program may be encoded by a number, thus it may be fed as input to a universal algorithm or function that may use it on any other input number. In other words, a Universal TM, U say, when given two inputs, the coding of a TM, A, and a number p, may simulate the computation that A would carry on p. This was derived from Gödel's fundamental idea of encoding the metatheory in the theory (of numbers), a key step in the proof of Gödel's incompleteness theorem for first order Arithmetic. In conclusion, in Turing machines, the (mathematical fiction) of hardware, the "head and tape", is distinct from the software, the "programs", as well as from the inputs, possibly numbers; however, they all coincide by coding the entire machine description by numbers [Davis, 1958].

The conceptual distinction between "hardware" and "software", as well as Gödel's encoding, are at the basis of today's computers and computers' programming. About ten years after Turing's mathematical description, early computers were even concretely designed by Turing himself and von Neumann (yet another great mathematician in this century, philosophically a formalist of Hilbert's school, who contributed to all main areas of Mathematical Logic and many of mathematics). Of course, the first thing to be said is that TM's, being abstract, are allowed to have an unbounded memory, the "tape", while actual computers are "finite state machines". However, this abstraction is one of the reasons of TM's up-to-dateness: any "a priori" limit set to memory in 1936, but even in ... 1986, would be ridiculous today. Most modern computers are still in the frame of TM's, conceptually; more precisely, they include an operating system and a compiler or an interpreter, i.e. a physical realization of a Universal TM. Their difference w.r.to clocks should be clear: it is due to the key distinction between hardware and software, as well as that auto-encoding possibility. In clocks (and Babbage machines) all feasible computations follow a predetermined algorithm, carved for ever in the material structure. The limited number of holes in Power's pieces of paper did not change essentially their nature as mechanical

devices. Today's programming languages, by formal, but "human-like" descriptions of the world, allow the simulation of all finitistically describable processes and realities into machines.

Or, at least, it seemed so and still seems so to many, as Turing and von Neumann immediately opened the way to the new metaphor of this century: mind (indeed, brain, for a monist) as Turing Machine[2].

The "so called" Turing Test [3] . In his seminal 1950 paper, Turing proposes a "imitation game" in order to discuss the question: <<can a machine think ?>>. Person C asks questions to a man A and a woman B, with the aim of discovering who is the woman. Then A is replaced by a machine. The claim is that, if rules fully govern human behaviour, we should be able to set up things in such a way that the machine may simulate A, ([Turing, 1950; §.8]). However, is there a finite set of rules for predicting human behaviour? Turing understands that a positive answer could lead to views analogous to Laplace's. Yet, he claims that the predictability he assumes <<is more effective>> than Laplace's, as Turing machines are discrete, while the Universe is not (§.5). And what about the brain? Turing acknowledges that neural machines need not be finite state machine. Yet, in a simulation game the difference - finite/infinite - may be undetectable, similarly as a continuous differential analyser cannot be distinguished, by a *finite* number of numerical questions, from a digital computer (§.7). The point then boils down to the assumption that humans have a <<complete set of behavioral laws>>, i.e. a finite and fixed set of rules. Cauciously, Turing only concludes that there is no scientific argument against this assumption (§.8). More audaciously, by assuming that these rules may be actually given, there began the adventure of (strong) Artificial Intelligence.

3. **Simulating Turing Machines by Clocks.**

Since so many, so great thinkers, for so long viewed mind as a clock, and so many so great still now propose TM's as a paradigm for mental activities, let's explore a new reductionist project for rationality: the reduction of TM's to clocks. Or, can we simulate a TM by a clock?

By such an investigation we may further elaborate on the distinction we hinted at between clocks and TM's: is this a purely quantitative (in the end, numerical) or a qualitative (conceptual) difference? In the first case, can we actually perform the tasks of a (universal) TM by taking enough clocks? And, what does the second alternative mean?

Of course, TM's are abstract mathematics (they have no limit on memory size), while clocks are concrete, finite, machines. So let's be more realistic and try to simulate actual

---

[2] See [Turing, 1950] and [Davis, 1995] for more on Turing, von Neumann, Logic and the brain; [McCorduck,1986] for a 30 years survey about this view.

[3] Turing makes little use of the word "test" and not in the usual sense of A.I., see [Lassègue,1994].

computers: it is not a potentially infinite tape that would make the difference, for our intended comparison, as mind has no infinite tape. Just take the largest, TM-like, real computer you need, or may conceive. As a computer is a finite state machine, it may then assume finitely many states, in a finitary language (i.e. based on a finite alphabet, possibly binary, made out of words of finite length). Thus, with enough clocks we may mimic all possible internal states (instantaneous descriptions, ID) of a large computer. It suffices to carve in steel all possible ID's and connect them along all possible paths, by wheels and pulleys, and... the reduction is performed.

In order to be more specific, consider a modern wonder, a Cray computer. Two years ago, the memory of a Cray was of about 64 M words of 64 bits. This is inscribed in chips of about 2cm size, each containing up to 8 Mega bytes. Assume, optimistically, that the binary information in a bit, may be encoded in a 1mm size binary mechanical device and then... compute. This is not the end though, as we only discussed memeory. Binary logic is implemented in chips containing up to 30 M ports. Moreover, the key point is that Cray are programmable. As they are finite state machines, though, Crays may compute only finitely many functions over a finite number of inputs, in contrast to TM's. Try now to mimic the previous memory and ports' structure, plus all possible, yet finitely many, computations as sequences of ID's, by a non-programmable mechanical device. Since the coding of a switch may require, at least, a 1mm square... then you will need more than the surface of our solar system.

There is a problem now. In Cray computers engineers have to optimize the internal design because of... the speed of light, or of electricity along the wires (about 1/3 of that of light or about 10 cm/nanosecond). The parallelism of these machines, due to a large number of concurrent and interacting processes, would be heavily affected by delays in communication. As a matter of fact, the interactions may be mostly sequentialized, by interleaving; thus, it is very relevant that exchanges between processors and memory, at the frequency of 100 MHz (roughly 100 Mips or 10 ns), do not happen at a distance larger than a few centimeters. On the grounds of Relativity Theory, it is easy to understand what it could mean if concurrent interacting computations are represented in different portions of the solar system....

Does this practical limit make a *qualitative* difference between clocks and Cray's or TM's, their formal counterpart? I believe so. Yet, let's put it in the following way. Assume that a "complexity issue" like this may still be considered only a *quantitative* difference and that it does not affect our philosophical understanding of the formal reduction we operated. In General Recursion Theory, for example, computability is unrelated to feasibility (see [Rogers,1967]), so that, by analogy, many may consider this reduction as "mathematically sound" and assume that the mechanicalism of rationalist reductionism may be equivalently based on Clocks or TM's. However, even if the quantitative reduction were "in principle" possible (from todays' programmable finite automata to clocks), its practical unfeasibility, by the increase in complexity, would be so high that a "qualitative jump" must be made in order to make it possible. The qualitative intellectual jump, in this case, is exactly what Turing proposed, by inventing programmable machines, based on a split between hardware and

software and on gödel-numbering of programs.

4. **Finite representations in physics and in metamathematics.**

In the previous section we tried to discuss how the "practical" issue of complexity, in reductions and simulations, may be so relevant as to blur the difference between "in principle / in practice". It makes no sense to say that, in principle, Crays (or the Macintosh I am writing on), may be reduced to clocks, as physical *principles* forbid it. However, the reader may still find a common basis for Clocks and TM's: they were and are meant as paradigms for stepwise computations over finite representations. More precisely, given a few << true and known principles >> , for Descartes, or an initial state of affairs, ID, deductions or the system proceeds by following a fixed and finite (decidable) set of rules. In particular, no further interference with the rest of the world is allowed. The other key assumption on which rationalist reductionism is based, is that any finitely representable process or piece of the world, including brain, should be completely describable in either formalism, Clocks or TM's. But what does "completely describable" mean? A way to understand it is given by the programs of Laplace and Hilbert. As already mentioned, they are strictly analogous: starting with a sufficiently accurate description of the initial conditions (a decidable set of axioms, for Hilbert), a finitistically presented (decidable) set of partial differential equations (rules of inference) allows full predictability (complete, decidable knowledge), up to a sufficiently accurate approximation of the output (there is no need for approximation in Hilbert's finitistic metamathematics).

We know now that three physical bodies, only subject to gravitation, or a forced or double pendulum (also the top oscillates), escape Laplace's predictability, by the well-known sensitivity to initial or border conditions: there may be no continuous dependence of the evolution of the system on the initial conditions [Poincaré,1892]. And yet, three bodies or a double pendulum, should be finitely representable: they occupy a finite portion of space, may assume finitely many states or positions (in view of quantum mechanics) and are subject to the simple laws of gravitation. Undecidability and incompleteness - see the Appendix - for systems including arithmetic (6 axiom schemata, instead of the initial conditions of three bodies or a double pendulum; three or four inference rules, instead of Newton's laws), are the corresponding negative results for the metamathematics of finitistic systems.

To summarize, in both cases, even closed “finite” systems are impredictable (undecidable, respectively), as the pendulum above or formal, first order, Arithmetic. The common lesson is that a few and simple deterministic rules, on finitary physical or logical structures, may give chaotic behaviors or very complex logical theories.

There is more though to be said about real mathematics. The existence of formally independent or undecidable sentences may be stated with no reference to meaning: it may be based on the construction of a formal sentence, provably underivable in a closed, finitistic logical system. However, the unprovability of a sentence wouldn't be so relevant if seen just

as the purely formal independence from axioms of a "meaningless" proposition, as a property of the intended *structure*. For example, Gödel's diagonal formula in PA, the first order formal theory of *numbers*, is meaningless, as a property of numbers[4]. It is *meaning* that steps in and gives the "essence" or significance of the negative results: what escapes the formalist, finitistic reduction, is "truth", as provability in the broadest sense, over the set of natural numbers, a very important structure in mathematics. This and other "structures" emerged in our natural and historical endeavor towards knowledge; they are determined and enriched by the unity of mathematics, as a growing and interconnected whole, where links and translations give understanding, suggest new methods of proof and give conceptual stability to invariants emerging from our relation to the world (such as the sequence of numbers, the group of symmetries in space etc.). In other words, the incompleteness theorems tell us that no fixed finitistic formal system can single out from the rest of mathematics structures as rich and relevant as the ones mentioned above. Numbers sit at the core of mathematics and reflect its unity and richness; by this, we are willing to accept proofs on them obtained by an ever growing variety of tools: Gödel's non-constructive proof of the independent sentence, transfinite induction, analytic methods, impredicative II order definitions, mechanical "follies" such as the proof of the four colour theorem (see the Appendix for more on proofs of unprovable sentences). The effectiveness and meaning of these tools may be given by different or new areas of mathematics (see [Kreisel&Macyntire,1982] for yet a different understanding of constructivity in Algebra).

In conclusion, the understanding of incompleteness, that we are trying to spell out here, is the following. We do have a precise notion of truth over the numbers (Tarski's, for example), as we can define what a Unicorn is, but we do not *have a hold* of true properties, nor of Unicorns. We can characterize, in abstracto, the true sentences, i.e. what it means for a sentence to be true, but we do not know it, individually, unless we prove it, in some *acceptable* way. We understand what is truth (or a Unicorn), but what we dynamically hold is "only" provability, in an open sense. In spite of the claim of Platonists, the more or less precise definition of truth for mathematical propositions or of Unicorns, and the failures of rationalist reductionism do not imply the existence of an Objective World, where mathematical truths and Unicorns live; they only encourage us to look for further proof methods.

Even though it is not yet clear how much relevant mathematics lies outsides finitistically provable fragments, the independence results hint to us that mathematics is an essentially open system of proofs, open to meaning, over an ongrowing variety of structures, and history. (See the Appendix for more on the proofs of unprovable propositions.)

---

[4] Gödel shows that deductions, as constructive inferences, are representable in PA and encodes, as a formula of PA, the metatheoretic assertion "this sentence is not provable in PA"; thus, by the very proof of its unprovability in PA, i.e. by finitistic or constructive methods, he gives a non constructive proof of its truth (see the Appendix).

5. **Mathematics as an open system.**

An actual deduction in mathematics, although a posteriori finitely representable, is not a formal game of rules in a fake laboratory of meaningless symbols, but lives by the ongoing interaction with other areas of mathematics and other forms of knowledge, not determined a priori. It is guided by the unity of mathematics, as a growing interactive system, as well as by a variety of cognitive experiences. Historical, live deductions are open to semantic analogies, which may suggest directions of work and conjectures. The a posteriori reduction of this interactive process to static formal systems devoid of meaning may be informative (meta)mathematics, but it is similar to the reduction of TM's to Clocks. Or even worst than that. Consider any relevant result in mathematics: it is rarely conjectured in a precise formal frame and then proved within it. Each step of the proof may require a truly relevant conjecture, a new general lemma. This may originate from an unexpected (possibly wrong) semantic analogy; it may use technical ideas from unrelated areas, which require a full change in the formal frame, or even language, used up to that point. The sensitivity of the result, which may also change in nature or extension along the way, to "bordering" meanings and to the dynamic process of proof, cumulates to the complexity issues related to the size of fully spelled-out formal proofs and which, by itself, resembles the TM/Clock translation. Mathematics, indeed each real mathematical proof, proceeds as an open system. The state of knowledge rarely is a predetermined set of axioms and rules: theories are built, axioms are modified and rules amended.

Consider, for example, a proof by induction. Induction is one of the clearest inference rules, in a formal approach. It looks like a fully mechanizable mathematical rule, as, at the inductive step, only a finite amount of cases need to be checked: once P(0) is proven, check whether P(0), P(1).... P(n) entail P(n+1), and deduce then P(n), for all n. However, every non trivial theorem, whose proof may be given by induction, is based on a non obvious "inductive load". That is, in order to prove P(n) for all n, one generally needs to find a stronger property Q, which implies P, as a way to derive Q(n+1) from Q(0), Q(1).... Q(n). The possibly infinite choice of such a Q is a major difficulty for automatic theorem proving, when dealing with relevant theorems [Dowek,]. The mathematician's discrimination and judgement depend on a very broad context, still to be explored.

In short, proofs grow and change by interaction with the rest of mathematics and of the world, via language, common sense and space descriptions. <<The [mathematical] proof changes the grammar of our language, it changes our concepts. It establishes new connections and creates the notion of these connections >> [Wittgenstein, 1956].

An investigation of this process, both as an historical one and as a local, one-proof related process, is still largely missing. The a posteriori, formal study of a proof or of an entire mathematical discipline is a different matter.

For example, given a new deep theorem of number theory or analysis, it is possible that there exists a constructive or predicative system that proves it. Constructive versions of number theoretic results often shed light on relevant results. In "reverse mathematics", some outstanding colleagues of ours (Simpson, Freidman and others) have been able to reconstruct

"backwards", predicative, indeed "least" axiomatic frames for a few fundamental theorems (see [JSL,1988]). This is extremely informative, by the generalizations it allows, and it is often very hard. It may even "justify" or found some branches of mathematics, for those who still feel that there is a problem about the "logical foundation" or formal roots of mathematics - as if mathematical logic, a branch of mathematics, could found the tree of which it is part (see [Wittgenstein, 1956]). The standard of rigor achieved today in definitions and proofs, also thanks to the work in logic, is *per se* a very robust "foundation" or basis. That approach says nothing, though, of the actual process of proof, the one that needs to be understood now or "found", by establishing its contextual dependencies from general knowledge and human experience.

The point is that each major result in mathematics originated, usually by its very proof, brand new theories or even tools. Assumptions need to be added or formal frames to be invented. Lebesgues' measure theory required the full use of impredicative definitions, as early as 1902, when their status was still very obscure. Gödels' incompleteness contributed to start recursion theory, originated the technique of Gödel-numbering, gave a first example of a provably non-constructive theorem. Gentzen established transfinite (ε_o-)induction, in order to obtain a consistency proof for PA; cut-elimination also stemmed out from his work as a new and fundamental technique. By Girard's work, it later gave rise to the non-constructive "candidates of reducibility", in a novel impredicative theory of proofs (II order lambda-calculus; see the Appendix).

Can all of this be generated in an initially fixed formal frame? In [Chaitin,1995], there is summarized a very interesting information theoretic analysis of incompleteness: finite sets of axioms may not contain enough information to derive a sufficiently complex theorem. A theorem then may be independent from a given theory, simply because it essentially contains more or extra information. Chaitin's work may show that randomness, as he claims, or the sensitivity to contexts or border/initial conditions is part even of elementary branches of number theory.

In conclusion, the invention of new and proof techniques, from broad contexts of knowledge, is part of the "theory building" and "tools building" so typical of mathematics.

It should be clear, though, that the formal analysis of proofs (current metamathematics) has had a major role: by examining results "in rigore mortis" or by the questions it raised, it gave us, in this century, a robust notion of mathematical rigor and many deep results (Lowenheim-Skolem, Gödel, Cohen's independence ... - by the way, mostly negative). Not to mention TM's and Crays, as we said above, since Computer Science originated from Logic. As a matter of fact, even the analysis of a frozen double pendulum, stopped two hours after it started oscillating, may be of some use: by splines, one may interpolate as many intermediate states as quantum mechanics allows and obtain, a posteriori, a wonderful, computer assisted picture of the path in the bifurcation which lead it to move clockwise or counterclockwise.

This critique of the analysis of deduction as a closed system is not meant to claim the death of metamathematics or of mathematical representations: on the contrary, in today's Logic and Theory of Computation, we should try to mimic the revitalization that Poincaré's results, the

KAM theorem or Ruelle's work brought to mathematical physics. Let's only hope not to be so slow: more than sixty years elapsed between Poincaré's three body theorem and the rediscovery of the issue by the russian school; probably a cultural blindness, due to the prevailing scientific paradigms. Indeed, it is a shame that as a student, in late sixties/early seventies, I was only taught Laplace's "rational mechanics", with not even a hint of what had happened ever since. This common experience to many, while students, is probably one of the cultural reasons for the still prevailing formalist and reductionist schools in Logic. I thus appreciated the apologies presented to the scientific community by Sir James Lightill, then chairman of the International Association for Mechanics : <<Here I have to pause and speak once again on the behalf of the broad global fraternity of practitioners of mechanics. We are deeply conscious today that the enthusiasm of the forebears for the marvellous achievements of Newtonian mechanics led them to make generalizations in this area of predictability which, indeed, we may have generally tended to believe before 1960, but which we now recognize to be false. We collectively whish to apologize for having mislead the general educated public by spreading ideas about the determinism of systems satisfying satisfying Newton's laws of motion that, after 1960, were to be proved incorrect>> [Lightill,1986][5].

The point, of course, is not to transfer the tools of modern mechanics in logic or theoretical computing, but, while emerging from Laplace/Hilbert cultural frame, to invent our own tools. This is being done, quite largely in the mathematics of computing, yet we need to understand better the dramatic change in perspective. The advantage is that today's challenge for us, logicians in theory of computation, more simply comes from the new, concrete, achievements in computers' design (which, so far, are simpler than Nature). The difficulty is that this requires an inversion of the historical paradigm, which so far went from Logic to computing.

6. **Interactive, asynchronous and distributed computing.**

In the previous sections, we tried to hint that the programs of Descartes, Laplace and Hilbert, namely the paradigms of rationalist reductionism and of modern science, share a common aim: the understanding of the world by closed systems of axioms and rules. A few negative, very surprising, results set the internal limits of this approach. In Mechanics, Logic and Computing, these results stress that there is no relevant closed and complete system. In this section we argue that even the current artificial systems for computing are not completely represented by rationalist reductionism.

As a matter of fact, in the last twenty years or so, computer design started to go beyond control of Proof Theory, which had originated it. By the progress of solid state physics and electronics, it became easy and cheap to have lots of processes going on at the same time and

[5] This remark about the little of attention to contemporary mathematical physics paid by logicians, who so long insisted on symbolic or functionalist reductions, is suggested by note 7, p.270 in [Putnam, 1989].

to let them interact with the rest of the world. This change dramatically accelarated in the last few years.

Interaction and on-line computations. Large computing systems, like airline reservations or banking systems, are based on a real time interaction with the environment. New information is inserted by autonomous agents; even rules may be modified by external agents or guarded commands, whose guards may depend on temporal as well as state sensitive firing conditions [Wegner, 1995(2)]. It is easy to prove that an "interactive Turing Machine", a machine which can read external inputs in real time, is strictly more expressive than an ordinary TM [Shœning,1995; Wegner,1995(1)]: an interactive machine is at least as powerful as an oracle TM (a machine that may ask during the computation whether a number belongs to an arbitrary set of numbers [Davis,1958; Rogers,1967]; for more on complexity of interactive systems, see [Goldwasser&al.,1985]). Further dynamicity is introduced by interactive interpreters. In LISP, for example (as in Universal TM's), the encoding of programs as data (the Gödel-numbering), allows the compilation of a program in LISP itself; or programs in LISP are "evaluated" (are given an operational meaning) within the language. More recently, in some Object-Oriented languages, this reflective property of interpreters has been extended by interaction. The interactive interpreters contain the specification not only of the system (axioms and rules), but also of an interactive device so that they can be modified by the environment. Engineers are (unconsciously?) approaching Wittgenstein's or Goodman's open understanding of rules.

Parallelism and concurrency. In modern computers, many processes are carried on in parallel. In some cases, these processes may be serialized and their mathematical descriptions may taken back to the classical TM's. Often though, massively parallel computing relies on concurrent processes, whose output depends on the occurrence of many events in a precise time lap: only the simultaneous occurrence, for example, of a certain number of events may cause a given effect[6].

Distributed systems. Computing systems may be distributed in space. Namely, independent computing devices may concur in a way that interaction among components is more expensive than internal communication. Distributed systems are usually organized in layers; a physical layer, the data links, networks, sessions, presentation and application layers [Wegner,1995(2)]. Each layer has its own conceptual representation problem; moreover they all affect each other while requiring different representations.

Asynchrony. One of the main problems with distributed computing is that of "time". Truly independent processors do not share the same clock. In a distributed system, possibly a large net of workstations, there is no notion of global or absolute time. Each processor has its own clock. There is no need to refer to relativistic effects to understand that their perfect synchronization is impossible: when individual operations are carried on in nanoseconds, slight differences in the measure of time heavily affect the ongoing computation. Not only in

[6] The relevance of the issue and the amount of literature is enormous, see [Lynch et al.,1994] for a recent text.

Einstein universes, but even in these artificial monsters we created and use everyday, newtonian time has lost its meaning. In the best cases, one may define an "average time", by non-obvious techniques from metrology.

A wide range of machines contain one or more of the previous facilities. A CM5, a modern Connection Machine, for example, may contain up to 1024 Sparc parallel processors (each with vectorial units) and reach 128 gigaflop (1 gigaflop = 1 billion operations per second). Clusters of workstations, with high speed network connections, may include many interacting computers, each open to human operators, who may modify data and programs, and each with its own clock. A large amount of practical and theoretical research is dedicated to the issues listed above. On the mathematical side, though, we are far away, for example, from the many deep results we have in the triangular relation between Proof Theory, Category Theory and Type Theory. These key areas of mathematical logic are at the base of functional programming languages, which are essentially sequential.

An analogy may help in understanding the new challenges we are facing in computing. Consider what is going on now in ... Paris. Millions of independent actions and events take place. Suddenly, M. Dupont is hit by a tile falling from a roof. The analysis of this event need not to be based on "chance". One may see the city as a distributed system of millions of parallel interacting asynchronous processes. The occurrence of a large number of them in a precise lap of time has caused the death of M. Dupont. Assume now that time is reversed and that everything starts over again exactly as it was 24 hours earlier. Laplace would have imagined M. Dupont killed once more. Poincaré, Arnold and Ruelle gradually taught to us that we may doubt of the necessary death of M. Dupont: the sensitivity of "Paris" to the initial or border conditions (the throwing of dices the night before, in a game, by the carpenter on the roof; a message, a few electrons, coming from outside Paris...). We may add to this the asynchrony of the ongoing processes. However, and also because of these reasons, time is irreversible and, up to now, no physical experiment could allow a double check on the fate of M. Dupont. Today we can. We can even decide to have or to avoid any (explicit) throwing of dices. In either case, it is a common experience to many computer scientists that, if a large distributed system of parallel interacting asynchronous processes is started over again with in the same initial conditions, the same rules and programs, one generally obtains a different output. Asynchrony first may modify the result, but also (minor) interactions with the external world, on-line data and program changes, reflective interactive interpreters may act like resonance in gravitational systems and discontinuously amplify indetectable variations on the initial or border conditions. It is worth noticing that Turing, in his 1950 paper, is concerned by the gap that there could be between formal and physical computability. The latter may be affected by the failure of Lapalce's predictibility for continuous systems, as <<one single electron ... could make the difference between a man being killed by an avalanche a year later, or escaping>>. As already mentioned, he believes that the difficulty does not show up, as it does in the case of <<differential analyzers>> (see above). Also von Neumann, while discussing his program towards an "electronic brain", stresses the

difficulties of formal descriptions, with reference to the complexities of measure in physical theories, as well as the inadequacy of Turing Machines for distributive processes ([Schnelle,1995]). As a matter of fact, a close analysis of the writings of these and other founding fathers, such as Descartes, Diderot, Laplace or Hilbert often gives a much broader and critical view than that of the followers'.

We are in a very stimulating moment for scientific knowledge: after sitting for centuries on the shoulders of giants, the fathers of modern science, and using the nowadays very clear, though difficult, tools of rationalist reductionism, the very machines which are the topmost applications of its developments force us to reconsider our methods. The grandchildren of Clocks, the Clusters of Computers, are no longer representable by closed systems of axioms and rules. The key point is that in their design and performance there is at least as much emphasis nowadays on communicating, sharing, coordinating ... as on algorithms. (For more references, see [Conrad,1995] on "evolutionary" styles of programming; [Hasslacher,1995] for parallel and dynamic systems; [Makowsky,1995] for fascinating remarks on the role of "discrimination" in selecting data and rules).

In order to obtain linguistic or geometric representations of the world and of these physical devices, reductions will still be required. But a broader notion of rationality is needed, based on systems that, as a software engineer aknowledges, are opened to a <<world which is only partially known and progressively revealed by interaction or observation>> [Wegner,1995(2)]. This fruitful crisis became evident in mechanics first, but its somewhat different extension to everyday computing, may more heavily affect Logic and common sense, and, thus, general scientific knowledge.

7. **Plasticity and the wet brain.**

We observed above that there is a qualitative jump in moving from the early mechanical paradigms of closed systems, clocks, to the closed systems of Hilbert and Turing, the sequential computers. Moreover, the depth and rigor of their mathematics allowed us to prove internally the limits, in expressive power, of these very approaches. Asynchronous, concurrent distributed systems, open to interactions, seem to suggest a further qualitative jump, whose mathematics is not yet fully understood.

Nobody seems to doubt that our brain is a massively parallel, distributed, interactive device, even though a few still try to reduce it to TM's and claim that, "in principle", any finite piece of the world should be fully describable by symbolic manipulations. As CM's seem to be more expressive, can we now deduce that electronic connection machines are the new metaphor for brain? The methodological doubt should first come to any scientific mind, since further qualitative jumps may still be needed. Let's list a few.

Dynamic connections. Neurophysiologists tell us that connections in brain change with time. Indeed, connections are stimulated by perception and psychological inputs: during the entire life span of an adult, not only in infants, stimuli affect changes in connections (a dramatic discovery, less than 10 years old). By comparing this fact to Turing's revolutionary

distinction between hardware and software and the gödelization of programs, one may view it as a further qualitative jump that takes us away from actual computing, where software does not concretely modify hardware, so far.

To be fair, this is exactly one of the features that our colleagues in neural nets try to mimic. They do it mathematically though, up to now. Namely, they do not have concrete machines whose hardware connections change according to software, but they can describe mathematically this process and simulate it numerically on large computers. Their mathematical merit is enormous, but it has a limit in complexity. Indeed, in CM's any node is potentially connected to any other node, thus changing connections may be surely simulated. The point though is that the circulation of signals turns out as a major programming challenge, since traffic jams make this dynamic handling of computations extremely difficult. And this is so in computers whose complexity, as for number of nodes, connections and operations per second are estimated to be between 1 ten-millionth to 1 millionth of the brain's complexity [Schwartz, 1989].

Informally, it is as if Turing, after inventing his mathematical abstraction, based on the distinction between hardware and software, hadn't been helped, 10 years later, by Von Neumann and electronics and, instead, had been forced to simulate his TM's (or the 1950 Eniac computer) over clocks. At the scale of that early machine with tubes and valves, this simulation could still be done, perhaps, but the qualitative jump of his idea wouldn't have displayed all its revolutionary value. Real neural nets, whose physical connections change dynamically, are probably as far from Turing Machines as these are from Clocks.

In view of the power of today's computers, the "equation", DynamicNets/TM's = TM's/Clocks, may seem excessive. I believe though that we should not be lead astray by the relevant mathematics in abstract neural nets, nor by the power of CM's: we are probably only at the beginning of the adventure of true neural net machines[7].

Convection in fluids. The plasticity of brain is also due to other fundamental phenomena. The complex blend, in vision, for example, between hologrammatic memorization of images and localization: certain brain lesions cause a general, but slight and not local, degradation of perception or image memory; others affect local reconstructions [Maffei&Meccacci,1979]. Moreover, the brain may often, though not always, substitute an area to another in carrying on a specific function; or may use the same area for implementing a substitutive function. The latter case is studied, for example, in [Bach Y Rita,1994], where tactile experiences may replace vision in blind people, probably by using some areas of the cortex usually dedicated to vision. Finally, it is not enough underlined that a lot of communication in brain goes by bio-chemical diffusion in liquids. Pain, in particular, seems to be transmitted by diffusion via surrounding fluids. This clearly gives an essentially different form of transmission: slow, unreliable, but in all directions [Bach Y Rita, 1994]. Matters may be yet more complicated as

[7] Berry, Vuillemin and many others currently work at the design of machines whose electronic connections change with time and programming. They can only handle, so far, machines with a few thousands nodes, in contrast to the 10 billion neurons of our brain.

the oscillations observed in exchanges of proteins between cells, may be chaotic [Goldbeter,1995] - oscillatory interacting reactions may behave like a double pendulum.

More aspects of brain physiology are listed in [Schwartz,1989], each of which may represent a qualitative difference, as the one between Clocks and TM's. All these features may be essential to brain functions: even if there seems to be redundancy in many individual functions, it is possible that the global functionality has no redundancy and that evolution has been optimizing matters, in those regards. In this case, the only simulator of brain and its functions would be an identical copy, embedded with a body into human history and society: the only way to acquire common sense and background knowledge, skills and practices. Yet, the quest for knowledge is a very good reason to continue the investigation. Moreover, in the the search for a deeper understanding, also by formal/computational models of specific functions, we may happen to construct better and better machines, with indirect fantastic fall-out. Man tried for long to simulate birds' flight. Human flight really began, stimulated by these attempts, when machines that parallel birds' flight were invented. We do not fly from branch to branch, today, but we can carry 50 tons at 1.000 km/h, a rather useful matter. Computers similarly parallel human reasoning. As for rigid wings, in computers (and human beings) rigid rules are not intelligence, but a substitute for it. The point now is to understand and scientifically describe the flexibility and multiplicity of intelligence, its "maze of background connections".

Both the analysis and the simulation of brain and its functions may require more mathematics, partly known, mostly yet to be invented. This may include recomposing, at an higher level, Turing's fruitful split between hardware and software, since the brain does not seem to implement it. As in the past, a change or a broader perspective in the philosophy of mathematics may possibly help.

8. **Implicit mathematics.**

We briefly summarized the historical itinerary that brought from the largely informal practice of Mathematics, to the current clear understanding of axioms, rules and rigor[8]. A crucial step was made when developing formal deductions into novel mathematics, the metamathematics of Frege and Hilbert; in other words, when the "explicit" part of mathematics was clearly defined and investigated. The final result of this itinerary has been such a clear display of mathematical rigor, that logical deductions could be implemented into fantastic machines for playing chess, making flight reservations, changing our clerical work

[8] See [Lakatos, 1976] for a description of the dynamic notion of rigor in mathematics; besides, it is amazing to browse in history and see how many wrong proofs were given in the past, by topmost mathematicians, from Fermat to Cauchy - often though these people had such a deep insight into mathematics that their theorems were true....

and everyday lives, except... for doing mathematics. Well, yes, there are several theorem provers; indeed, a few are even very effective, provided that... they are not left alone, even in inductive proofs (see §.5 above). Only interactive programs achieve some results. Why the linguistic, explicit rules of mathematics do not suffice to do mathematics? The formal, linguistic clarification has excluded, to say the least, the role of direct geometric manipulation, the open growth of the mathematical practice, the overall unity of mathematical (and human) knowledge. In a word, it ignored what may be called "implicit" mathematics and its background knowledge.

In this century, many mathematicians tried to blend various forms of knowledge and historical or practical experiences into mathematics. It may suffice to recall the many remarks or writings by Poincaré, Hadamard, Polya, H. Weyl, Enriques.... Their perspective was little developed though, in view of the successes of the formalist approach and because they did not fully succeed, except, in part, Poincare' or Weyl (see [Longo, 1989]), in turning their psychological remarks into a philosophy of mathematics.

The report of a personal experience has been often used in those reflections, and it may actually help, by the passion each researcher has for his own results. Let's immodestly dare to do so here. I recently collaborated on a result in my research area (by the way, a modern development of Church's formal language of functions). Some observed that it is an elegant theorem. What does elegance mean? It is a matter of "informal" symmetry, of similarities with properties of holomorphic functions, which were totally unrelated. Or, perhaps, a clear intuitive understanding of computations on sets, the one line statement, logically and "aesthetically" heavier on the left, lighter on the right... I cannot distinguish these judgements from the criteria by which we may appreciate a painting or a handcraft. The conjecture had been made possible by an improper reference to an informal model, a reference based on a contamination of languages, where in an earlier version of the proof we confused categories, sets and proofs. The "a posteriori" display of the "formal" steps we made is surely incomplete and their full TM formalization would be comparable to the... carving of steel clocks. Besides the (feasible?) a posteriori simulation, the *conjecture* is still missing. But, also the proof technique and the frame language needed to be conjectured as well as the right lemmas to be singled out. Where do "implicit" judgements based on elegance, analogies with unrelated intellectual experiences stop (the musical/geometric role of a zig-zag we have in the proof has been clear to us) and formal matters begin? There is no magic here; or, more exactly, we should stop considering magic or not a subject of investigation, what is not captured by axiomatic methods and may require a broader analysis of mathematical knowledge. We should investigate its unity with other forms of knowledge or, even, stress, with Diderot, that there is no knowledge without feelings, joy and sufferings.

Hardy insisted in several places on the purposeless esthethics of mathematics [Hardy,1940]. His views, which are not those expressed in this paper, are still now widely accepted among mathematicians and were meant to remove the mathematical work from rational investigation and practical aims. We should instead bring under the realm of scientific knowledge also the skills and discriminations in the practice of mathematics.

Some may say that we are confusing here the (formal, logical...) "justification" or "foundation" with the "context of discovery". Indeed, this is exactly what we need to reconcile, as that distinction is by now very well understood, in particular by the work in metamathematics, which was meant to found on certainty and rigor the existing mathematics. The mistake instead is the continuing use of formal tools, invented to "justify" and set a standard of rigor, as tools for the analysis or even the progress of the mathematical discovery (see, for example, [Michie,1995] on the Japanese V generation of computers or [Beeson, 1995] whose slogan still is: "mathematics = logic+computation").

Since the founding fathers largely clarified for us explicit mathematics, we need now to go further on: we need to recompose knowledge and embed mathematics into the maze of back connections which relate it to perception, judgements and other forms of intelligence. These recomposition may pass by the understanding of a large varieties of individual, physical and historical experiences. In this century the way has been opened by H. Poincare' and H. Weyl, as already said, though from different perspectives. Further references and a few hints, concerning memory structuring, space and images, linguistic and geometric invariants, may be found in [Longo,1992,1995,1996].

The difficulty is to turn scattered remarks into a foundation of mathematical *knowledge*(in contrast to the "internal" foundation of mathematics), by seeing mathematics as a discipline not transcending human experience, but emerging from it. For example, we need to explore the cognitive process that makes a mathematician claim that he "sees" a many dimensional differential variety (<<mathematics... whose main organ is "seeing">>, as Weyl suggests) and single out the mental invariants and images underlying this insight. Yet, such an understanding should not be an isolated psychological remark, but part of a philosophical proposal concerning knowledge.

To this aim, we should work at least in two directions. In short, we have, on one side, to use the rigor and generality of mathematics in order to design or describe the open systems of computations and life (mathematics as a tool). On the other, we should investigate actual mathematical reasoning as a broader paradigm than so far done (implicit mathematics as an object of study). In view of the relevance of the example set by mathematics for general reasoning and for the invention of metaphors and machines, this may help in computer design, cognition and their philosophy.

Acknowledgements. I would like to thank Peter Wegner for many stimulating discussions during the LICS'95 Conference, in San Diego. Kim Bruce, Gilles Dowek, Jean Lassègue and Lorenzo Seno made several comments, with a critical insight. Philippe Matherat and Roberto DiCosmo helped with comments and data on large computers.

APPENDIX: More on the proofs of unprovable propositions.

In a footnote we already hinted that the truth - over the standard model of first order Number

Theory, PA - of Gödel's unprovable proposition, let's call it G, is proved by the very proof of its unprovability in PA (the first incompleteness theorem) and by the second incompleteness theorem. G says "this sentence is not provable in PA", then, if PA is consistent, i.e. non-contradictory, G cannot be proved in PA and, at first glance, this proves that it is true. Yet, this argument should be somewhat spelled out. The second incompleteness theorem is based on formalizing, within PA, the metatheoretic notion of coherence of PA, i.e. on giving a proposition of PA, Coher, whose "meaning" is that PA is coherent (basically, Coher says that PA does not prove "0=1"); then the theorem proves that, in PA, one may prove that Coher and G are equivalent. In particular, then, Coher implies G. Thus, if one assumes Coher (i.e. that PA is coherent), than, as Coher implies G even formally (and easily: the second theorem is "easy"!), one deduces the truth of G. In a sense, the second incompleteness theorem has two consequences: first, and mostly, it shows that Coher is unprovable in PA, second, as a corollary, it gives a proof of the truth of G, with no handwaving. However, one *needs* to assume the coherence of PA in order to *prove* that G is true. Any "direct" proof of the truth of G, would immediately give proof of coherence of PA: this can be done, in a non finitary fashion, but it is not easy. In conclusion, one should never say then that G "is true", like magic: it is easily proved, under assumption of coherence; it is also shown with no assumption on Coher, but outside PA and by the many difficult proofs of coherence of PA. In summary, the "truth" of G *must* be proved, but it cannot by derived by effective tools, unless assuming coherence, nor by a stepwise construction nor as a recursive function as these equivalent concepts are all expressible in PA (Gödel's key representation lemma of the finitistic, effective metatheory in the theory).

We next discuss some other cases of proofs of constructively unprovable sentences. Let's first recall a common practice in mathematics concerning universal statements, which are important both in everyday mathematics and in theory of proofs. The discussion is slightly more technical.

Generic Proofs Given a mathematical theory which allows universal quantification, i.e.sentences such as $\forall x.P(x)$, how does one prove them? If the range of quantification is infinite, there is no way to explore and check each individual case. The practice of mathematics is usually the following: check what is the "intended range of quantification", i. e. over which set, domain or structure the universally quantified x is meant to be interpreted (the set of real numbers, for example, or any other countable or *uncountable* structure). Then prove $P(a)$, where a is an arbitrary or *generic* element of that domain ("take a to be a real", typically), that is give a proof of $P(a)$ where *the only property of a, used in the proof*, is that a belongs to the intended domain. In Type Theory we would say: only the *type* of a is used in the proof. By this, the proof of $P(a)$ is a prototype or paradigm or pattern for the proof of $P(b)$, for any other b in that domain. Thus, one has a proof of $\forall x.P(x)$, i.e. a proof that $P(x)$ holds everywhere in the intended domain of interpretation of

x.

Consider now the special case when the (intended) domain of interpretation is the set of natural numbers or any countable well-ordering. How does this technique relate to induction? The implicit "regularity" of a generic proof - all proofs are given by the same reasons relatively to a given formal frame - may be not present in induction. Once proved P(0), which may be ad hoc, one proves, in an inductive argument, the *implication* $P(n) \Rightarrow P(n+1)$. This is the proof that, soon (in general, at the first level of the application of the rule) or late (in case of nested induction), must be generic in n. Note also that this proof is based only on the *assumption* of P(n) *not* on its proof (as for *derivable* vs *admissible* rules in logic) So, formally, P(n) and P(n+1) may be *true for different raisons* or their individual *proofs* may follow different patterns and yet, the proof of $P(n) \Rightarrow P(n+1)$ may be generic in n.

From generic to specific proofs. In the same period as the consistency of PA was shown, by Godel, to be unprovable in PA, Gentzen gave a proof of consistency for PA. Clearly this proof cannot be given in PA itself. A key technique in Gentzen's approach is called "cut-elimination". The idea is the following. A proof of a proposition Q is *direct* when it contains no sentence "more complex" than Q, where more complex refers to the logical/syntactic structure of Q, e.g. the number of quantifiers. However, in mathematics and its theory of proofs, there are plenty of indirect proofs; typically, one may use a general lemma, e.g. a proof of $\forall x.P(x)$, in order to prove a special case, P(a), which may happen to be part of Q. As well known, these kinds of general lemmas are widely used, since they may be given by a generic proof, as above. Cuts are eliminated when one gets rid of these detours and directly proves P(a). By this, a proof becomes direct or it is turned in a fully known and unique form, a "normal form". To show that one can eliminate all cuts is hard and not formalizable in PA. Indeed, "cut-elimination" implies, *within PA*, the consistency of PA, which is unprovable. This is the formal reason for the unprovability of "cut-elimination"; the actual reasons are spelled out in [Girard&al.1989], where Tait-Girard's technique for cut-elimination, called "the candidates of reducibility", is used. In short, given a set of formulae A in a theory T (e.g. Gödel's T), and a set Red(A) of candidates of reducibility relative to A, the crucial statement "t is in Red(A)" can't be proved uniformly for all t and A or by using the same formal paradigm for all of them, in the language of T. As a matter of fact, its very formalization in T has a different logical complexity (the quantifiers alternate differently) depending on A. Thus, in particular, no generic proof in t and A, in the sense above, applies, within T. This is why, even though each individual statement is provable in T, the intended universal quantification is not.

Paris-Harrington theorem and the type of a generic proof. A recent and interesting unprovable sentence of PA is given in [Paris&Harrington,1978]. In contrast to Godel' diagonal sentence, the proposition is interesting as a "concrete" property of numbers, since it is a "minor" variant of a well know and relevant property due to Ramsey (the finite Ramsey

Theorem, see [Drake,1974]). The finite Ramsey Theorem is provable in PA, while Paris and Harrington showed that their sentence, call it $\forall x.PH(x)$, implies, within PA, the consistency of PA. There is an alternative way to prove the unprovability of $\forall x.PH(x)$, based on work on models of arithmetic by Paris and Kirby: one may give, for any non standard model of PA that realizes $\forall x.PH(x)$, another non standard one that does not realize it, see [Berestovoy&Longo,1981]. The point though is that $\forall x.PH(x)$ is true, over the numbers, i.e. in the standard model. It is much easier to show the truth of $\forall x.PH(x)$ than its unprovability, but it is truth that interests us here, as we want to look at its non constructive proof and at the role of generic proofs. We just mention the key steps.

For each number n one can show that $PA \vdash PH(n)$, i.e. that PA proves $PH(n)$. The argument is "per absurdum": fix an arbitrary (generic) number n and assume that there is a model M which does not realize $PH(n)$. Then, by using König's Lemma and an infinite version of Ramsey Theorem, one gets to a contradiction. Both the lemma and theorem are non-constructive: the first says "in a finitely branching infinite tree there is an infinite branch". This seems obvious, but there is no way to obtain uniformly effectively the infinite branch. The second is a more complex infinitary statement, see [Paris&Harrington, 1978; Berestovoy&Longo,1981]. In the end, though, we have, for all n, $PA \vdash PH(n)$. That this, per absurdum and via a non-constructive argument, one shows that there must exist, for each n, a constructive proof of $PH(n)$, without providing it explicitly.

Note now that the number n is used generically, in the proof. Why can't we go from it to a proof of $\forall x.PH(x)$? The point is that n is used in the proof as a natural number, i.e. the proof refers to its intended "type" as that of the standard model (its property of being a number is used both in the finite branching of König's tree and in the application of Ramsey Theorem). While first order arithmetic, PA, has also non standard models: thus, x in $\forall x.PH(x)$ is meant to range on the domain of all possible models. In contrast, then, to the previous case (the candidates of reducibility), we do have a general paradigm or a generic proof, within PA, but this paradigm is restricted to a specific domain of interpretation of x. The unprovability of $\forall x.PH(x)$ proves that this restriction, in the generic proof, is unavoidable. (Moreover, there is not even the possibility to express this restriction by a bounded quantification and prove $\forall x.(N(x) \Rightarrow PH(x))$, as there is no way to define a formal type or predicate N of the natural numbers in PA.)

In conclusion, we briefly reviewed two relevant cases of true but unprovable theorems. There is no metaphysics, though, no intended platonistic reality: their proofs simply escape the limited frame of finitism, as they cannot be given in PA (and this is provable), yet their truth is proved. In both cases, a typical technique in mathematics is used (a technique not enough discussed in logic), that of generic proofs. In either case, the generic argument cannot be extended to a proof of the universally quantified assertion in PA, but for different reasons: a lack of logical uniformity, as for "candidates of reducibility", the specific type of

the generic variable n, as for Paris-Harrington proposition. Clearly, both proofs can be "a posteriori" fully formalized, in a non finitistic fashion: take (II order, impredicative) ZF or similar formal set theory, extended exactly as to handle transfinite or ε_0-induction (i.e. assuming the existence of a large enough cardinal), and that is all you need, once you know the proof.

References

Bach Y Rita P., **Memosynaptic diffusion neurotransmission and late brain organization**, 1995.

Buffat M., "Diderot, le corps de la machine", **Revue des Sciences Humaines**, n. 186-187, 1982.

Beeson M. J., "Computerizing Mathematics: Logic and Computation" *in* [Herken,1995].

Berestovoy B., Longo G., "Il risultato di indipendenza di Paris-Harrington per l'Aritmetica di Peano", GNSAGA, Roma, 1981

Chaitin G. J.., "An Algebraic Equation for the Halting Problem" *in* [Herken,1995].

Conrad M., "The Price of Programmability" *in* [Herken,1995].

Davis M., **Computability and Unsolvabilty**, MacGraw-Hill, 1958

Davis M., "Mathematical Logic and the Origin of Modern Computing" and "Influences of Mathematical Logic in Computer Science" *in* [Herken,1995].

Descartes R., **Regulae ad directionem ingenii**, 1619 - 1664.

Diderot D., **Le rêve de d'Alembert**, Flammarion, 1973.

Dowek G., **Démonstration automatique dans le Calcul des Constructions**, Thèse, Univ. Paris VII, 1991.

Drake F., **Set Theory**, North Holland, 1974

Enriquez F., **Problemi della Scienza**, Torino, 1909.

Enriquez F., **Natura, Ragione e Storia**, Antologia di scritti, Einaudi, 1958.

Girard J.Y., Lafont Y., Taylor P. **Proofs and Types**, Cambridge U.P., 1989.

Goldbeter A., "Non-linearity and chaos in biochemical reactions", lecture delivered at the Academia Europaea, Cracow, 1995.

Goldwasser S., Micali S., Rackoff C., "The knowledge complexity of interactive proof

systems", **17th ACM Symp. on Theory of Computing**, New York, 1985.

Goodman N., **Fact, Fiction and Forecast**, Harvard U.P., 1983.

Graubard S. (ed.), **The Artificial Intelligence Debate**, M.I.T. Press, 1989.

Hadamard J., **The psychology of invention in the mathematical field**, Princeton U.P., 1945.

Hardy L., **A mathematician apology**, 1940.

Hasslacher L., "Beyond the Turing Machine" *in* [Herken,1995].

Herken R., **The Universal Turing Machine**, Springer-Verlag, 1995.

Hoffmann P., "Modèle mécaniste et modèle animiste", **Revue des Sciences Humaines**, 186-187, 1982.

Kreisel G., A. Macintyre "Constructive Logic versus Algebraization I" *in* **Brouwer Centenary Symposium**, North-Holland, 1982.

JSL, Feferman S., S. Sieg, S. Simpson, three papers dedicated to the partial realization of Hilbert's program, **Journal of Symbolic Logic**, 53, 2, 1988.

Lakatos I., **Proofs and Refutations**, Cambridge U. Press, 1976

Lassègue J., **L'intelligence artificielle et la question du continu**, Thèse, Univ. Paris X - Nanterre, 1994.

Lightill J., "The recently recognized failure of predictability in Newtonian dynamics" **Proc. R. Soc. Lond.**, 407 (35-50), 1986

Longo G., "Some aspects of impredicativity: notes on Weyl's philosophy of Mathematics and on todays' Type Theory" **Logic Colloquium 87**, Studies in Logic (Ebbinghaus et al. eds), North-Holland, 1989 (downloadable by ftp).

Longo G., "Reflections on Mathematics and Computer Science", **European Congress of Mathematics**, Paris 1992; (Ebbinghaus ed.), 1994 (downloadable by ftp).

Longo G., "Sur l'émérgence de l'objectivité des Mathématiques", **Intellectica**, à paraître, 1995.

Longo G., "The continuum: foundations and applications", en préparation, 1996

(downloadable by ftp).

Lynch N. et al., **Atomic Transactions**, Kaufmann, 1994.

McCorduck P., **The Universal Machine: Confessions of a Technological Optimist**, McGraw Hill, 1986.

Makowsky Y.A., "Mental Images and the Architecture of Concepts" *in* [Herken, 1995].

Maffei L., Meccacci L., **La Visione**, Mondadori, 1979.

Paris J., Harrington L., "A mathematical incompleteness in Peano Arithmetic", **Handbook of Mathematical Logic**, 1978.

Poincaré H., **Les Méthodes Nouvelles de la Mécanique Celeste**, Paris, 1892.

Poincaré H., **La Science et l'Hypothese**, Flammarion, Paris, 1902.

Poincaré H., **La valeur de la Science**, Flammarion, Paris, 1905.

Putnam H., "Much Ado About Not Very Much", *in* [Graubard,1989].

Rogers H., **Theory of Recursive Functions and Effective Computability**, 1967.

Turing A, "Computing Machines and Intelligence", **Mind**, LIX, 1950.

Schwartz J., "The New Connectionism", *in* [Graubard,1989].

Shœning U., "Complexity Theory and Interaction" *in* [Herken,1995]

Wegner P., "The expressive power of interaction", Brown Univ. Techn. Rep., 1995.

Wegner P., "A framework for empirical Computer Science", Brown Univ. Techn. Rep. CS-95-18, 1995.

Weyl H., **Das Kontinuum**, 1918.

Weyl H., **Symmetry**, Princeton University Press, 1952.

Wittgenstein L., **Remarks on the foundation of Mathematics**, 1956.