News from New Zealand

BY

C. S. Calude

Department of Computer Science, University of Auckland
Auckland, New Zealand
cristian@cs.auckland.ac.nz

1 Scientific and Community News

0. The latest CDMTCS research reports are (http://www.cs.auckland.ac.nz/staff-cgi-bin/mjd/secondcgi.pl):

375. M.J. Dinneen, Y.-B. Kim and R. Nicolescu. P Systems and the Byzantine Agreement. 01/2010

376. M. Andreev, I. Razenshteyn and A. Shen. Not Every Domain of a Plain Decompressor Contains the Domain of a Prefix-Free One. 02/2010

377. Y.I. Manin. Renormalization and Computation II: Time Cut-Off and the Halting Problem. 02/2010

378. M.J. Dinneen, Y.-B. Kim and R. Nicolescu. Synchronization in P Modules. 02/2010

379. V. Putz and K. Svozil. Can a Computer be "pushed" to Perform Faster-Than-Light? 03/2010

380. K. Tadaki. A New Representation of Chaitin Omega Number Based on Compressible Strings. 04/2010

381. A.A. Abbott and C.S. Calude. Understanding the Quantum Computational Speed-up via De-quantisation. 04/2010

2 A Dialogue about *Qualitative Computing* with Professor Françoise Chatelin

Professor Françoise Chatelin `http://www.cerfacs.fr/~chatelin` *teaches Applied Mathematics at the University Toulouse 1- Capitole, and is the head of the Qualitative Computing group at the Centre Européen de Recherche et de Formation Avancée au Calcul Scientifique (Cerfacs) in Toulouse, France. Her expertise includes many areas, from spectral theory for linear operators in Banach spaces to finite precision computation of very large matrix eigenproblems. Professor Chatelin has supervised 31 Ph.D. theses and has authored three books which are now classic references; the fourth one, to appear at World Scientific, Singapore, is discussed here. Before moving to Toulouse, Professor Chatelin taught at the Universities Grenoble 2 - Pierre Mendes-France and Paris 9 - Dauphine. She has been a visiting researcher at Berkeley and Stanford Universities, IBM San Jose (Ca) and Yorktown Heights (NY). For almost a decade (from 1984 to 1993) she was a scientific manager (in charge of intensive computing) at the Centre Scientifique IBM-France in Paris and the Laboratoire Central de Recherches Thales near Paris.*

Cristian Calude: Your latest book (soon to appear at World Scientific) develops a theory of computing—which you call qualitative computing—using general multiplicative algebras. What is qualitative computing?

Françoise Chatelin: Qualitative computing is a branch of mathematics which extends analysis and algebra over $\mathbb{R}$ and $\mathbb{C}$ by specifically looking at how the laws of *classical* computation (Euler-Cauchy-Riemann-Jordan-Puiseux) are modified when mathematical computation does not take place over a *commutative field*. This fills a gap since most college-level textbooks in mathematical analysis only consider numbers which are either real or complex. And modern abstract algebra is *not* driven by computation.

CC: So we are talking about numerical computation. Why classical numbers are not enough?

FC: Numerical in a broad sense, where “numbers" are defined as entities over which well-defined computation can be performed on. There are important practical domains where classical numbers are too *limited*. Let me cite very different examples.

In physics, the quaternions which form a *non*-commutative field of numbers with four real dimensions are the language of Maxwell's electromagnetism and special relativity. In computer graphics, they are essential for 3D motion pictures.
Another example is found in the booming field of numerical linear algebra: to speed-up computation, the basic "numbers" are often taken to be square matrices from an associative *algebra* (over $\mathbb{R}$ or $\mathbb{C}$). This is essential for large computer simulations required by high tech industries.
The third example is better known: it consists of vectors (or strings or sequences) defined over a finite ring of scalars. The scalars $\{0, 1, 2, 3\}$ in $\mathbb{Z}/4\mathbb{Z}$ have a ring structure with 2 as a zero-divisor.

CC: In the first two examples multiplication seems to be associative too. What is the reason for dropping associativity altogether?

FC: To favour *recursiveness*, which enables complexification and creativity, two properties which are ubiquitous in life's phenomena.

CC: Why non-associativity is better suited for recursiveness and how this feature "enables complexification and creativity"?

FC: I would rather say that the recursive definition of the 3 algebraic operations, addition, involution, multiplication, when applied to real vectors in $\mathbb{R}^{2^k}, k > 0$, implies that multiplication becomes necessarily non-associative for $k \geq 3$. This does not happen when $\mathbb{R}$ is replaced by $\mathbb{Z}_2$, giving birth to binary sequences of length 2^k. Then involution reduces to the identity map, and multiplication remains commutative and associative. Complexification is defined below, and creativity is also related to paradoxes to be presented later.

CC: Is such a recursive definition for multiplicative algebras a new idea?

FC: Not at all: the idea—which goes back to the American mathematician Dickson—is about a century old! Originally, Dickson was looking for an algorithmic way to derive the multiplication table for the (non-associative) octonions $\mathbb{G}$ of Graves (December 1843), from that for the (non-commutative) quaternions $\mathbb{H}$ of Hamilton (October 1843). This lead him to the discovery in 1912 of the recursive doubling process which defines an unbounded sequence of multiplicative real algebras A_k (over $\mathbb{R}$), $k \geq 0$ in $\mathbb{N}$. Let 1_k (resp. $\tilde{1}_k$) represent the real (resp. complex) unit in A_k, then

$$A_0 = \mathbb{R}, \;\; A_k = A_{k-1} \times 1_k \oplus A_{k-1} \times \tilde{1}_k, \quad k \geq 1,$$

where $1_k = (1_{k-1}, 0)$ and $\tilde{1}_k = (0, 1_{k-1})$ satisfy $\tilde{1} \times \tilde{1} = -1$. Starting from $A_0 = \mathbb{R}$, one gets successively $A_1 = \mathbb{C} = \mathbb{R} \oplus i\mathbb{R}$, $\tilde{1}_1 = i = (0, 1)$ for $k = 1$, and $A_2 = \mathbb{H} = \mathbb{C} \oplus \mathbb{C} \times j$, $\tilde{1}_2 = j = (0\,0, 1\,0)$ for $k = 2$.

In the complexification process where A_{k-1} yields A_k, the complex unit $\tilde{1}_k$ is constructed by assimilation of a vector foreign to A_{k-1}. Therefore each "complex" algebra A_k possesses features which are not present in its "real" part A_{k-1}. This is an aspect of creativity resulting from complexification, nonlinearity and induction ($k-1 \longmapsto k$). This vastly differs from the much simpler creative process known as mathematical induction. In Dickson algebras multiplication displays new features for each $k \geq 0$. By way of contrast, let us consider associative Clifford algebras C_k (over $\mathbb{R}$) used in high energy physics. Their structure obeys the evolution law with period 8 given by $\mathbb{C}_{k+8} \cong \mathbb{C}_k \otimes \mathbb{R}^{16\times 16}$ (Cartan, 1908). The sequence C_k is completely determined by the eight first algebras C_0 to C_7.

Vectors in A_k, $k \geq 2$, have been called *hypercomplex numbers* (Hurwitz, Dickson). Accordingly, computation in A_k, $k \geq 2$, was called *hypercomputation*. The original idea of Dickson was later developed by considering scalars in other algebraic structures than $\mathbb{R}$. If one is interested in the various transformations of multiplication as the dimension 2^k of the numbers increases, then the case of scalars in a finite ring (leading to discrete mathematics) is no less important than that of real scalars (leading to continuous mathematics).

CC: Can you briefly describe the pros and cons of computing with elements in a real Dickson algebra A_k?

FC: Hard to be brief because each A_k, $k \geq 3$, has specific properties which makes it unique in many respects. Any math student knows that analysis over $\mathbb{C}$ differs greatly from analysis over $\mathbb{R}^2$, despite the isomorphism $\mathbb{C} \cong \mathbb{R} \times \mathbb{R}$. This is but the simplest form of an underlying analytical engine in A_k, $k \geq 1$, which takes a specific form at each k.

As long as multiplication remains associative ($k \leq 2$), nonlinear computation in A_k remains classical and yields the absolute certainty that mathematicians are accustomed to, the very certainty that singles out mathematics from all experimental sciences. The situation changes drastically in the absence of associativity for $k \geq 3$.

In a nutshell, non-associativity induces measurement paradoxes which modify the local geometry and challenges classical logic. Paradoxes signal a clash between the global nonlinear viewpoint and the local linear one. It turns out that paradoxes and freedom of choice between several computational routes can pop up anywhere. In non-associative algebras, there only exist competing answers which are all tentative. Their validity becomes *relative* to a backward analysis test elaborated by the computing agent. Therefore the special status of classical mathematics does not hold without associativity. Maths becomes *experimental*: it tells what "is possible" and no more what "it is." The analysis based on paradoxical computation stops being deterministic *without* becoming random.

CC: Please describe a measurement paradox?

FC: A measurement paradox is related to the Singular Value Decomposition (SVD) of the multiplication map. In dimension 16 and higher, a vector defining a multiplication has several measures, with only one agreeing with Euclid's norm. Even more strikingly, the Euclidean norm of a non-zero vector may be computed as 0; hence Euclidean measurements may not be reliable. As I said earlier, there may be several different answers to the same question, depending on the chosen computational route. The two contradictory answers 0 and 1 may well be valid computational outputs, showing the limits of classical logic. Such a paradoxical phenomenon evokes the celebrated measurement problem in quantum mechanics. What looks like a logical nightmare can be interpreted as a blessing: the freedom to choose between several computational routes favours creativity. But extreme caution is necessary, since nonlinear computation plays havoc in classical analysis when it becomes paradoxical.

CC: So, what do we compute?

FC: I assume that your question is about any sort of computation, as general as we can imagine it today. If we talk about conscious computing such as arithmetic and beyond, we should keep in mind that, if all human groups have developed language skills, not all count beyond 1, 2, many. So conscious counting is not necessary for survival... Computation which sustains life in organisms seems to take place mainly at an unconscious level, in which measurements play an important role because the flow of information has to be delicately balanced. Dreams often express a sort of symbolic computation within the human psyche.

But I guess that your question is more concerned with our technology-based society. For millennia, computing has been the driving force behind the development of mathematics. However in the twentieth century, it has been used mostly to develop the techno-science, the science only driven by technology, for which classical mathematics is good enough. Moreover the swift computerisation of our society is fuelled by the impressive feats of the computer science community, which receives an invaluable help from numerical software developers. It may not be widely known that part of the worldwide success of Google should be credited to Gene Golub. This leading figure in numerical linear algebra was a very influential professor at Stanford for almost half a century. He was also the scientific advisor for many numerical routines developed to create the 1st-rank search engine which lead to the current supremacy of Google.

On the life side in the twenty first century, Nature and lab experiments both indicate limitations. If we want to progress in our understanding of *life* beyond the simplistic picture that it emerges "in principle" from physics and chemistry, it may be wise to look seriously at the amazing properties of paradoxical computation.

CC: Which agents compute? Does Nature compute?

FC: Because mathematics is a creation of the human mind, it seems hard to maintain that other living beings are endowed with a man-created ability! However the celebrated "unreasonable effectiveness of mathematics in the natural sciences" (Wigner, 1960) indicates that scientists can safely reason *as if* they did, as if Nature had the ability to process information by mathematical computing. But they should never forget that any human explanation by a theory is anthropomorphic: it may be a far cry from "natural reality."

CC: What is the goal of computing for human beings?

FC: This is a vast question which is a matter of much debate between psychologists and neuroscientists. And there is today no consensus about what "computing for human beings" could mean "in reality." For the sake of this dialogue, let us assume that man-created computation in the broadest sense is a plausible model for some of the uncountably many ways by which the human mind can process information. I suggest that, deep down, we compute in an effort to understand life as we experience it, both in the world outside and in our own inner world. We try to order the world according to reason. The goal is to help building our *imago mundi*, that is an image of the world specific to each of us, and necessary to navigate in it. In my book, I postulate that hypercomputation is an important tool to explore the mind's processes which appear to us as nonlinear computations. This shows in the subtitle of the book: "A computational journey into nonlinearity." An essential epistemological tool for the construction in this specific context, is the notion of *algebraic irreducibility by linear derivation*, which characterises the limits of any explanation by *linear* causality within an inherently *nonlinear* algebraic context when $k \geq 2$. The dimension of the irreducible nonlinear core of A_k is by definition the algebraic depth of A_k (over $\mathbb{R}$). It can be shown that the three division algebras $\mathbb{R}$, $\mathbb{H}$ and $\mathbb{G}$ have an algebraic depth equal to 1; all other A_k, $k \notin \{0, 2, 3\}$, have an algebraic depth ≥ 2.
The algebraic depth is a measure of *algebraic* complexity, not to be confused with the *descriptive* complexity (measured by the size of a computer program) in computer science. Let me add that the extended commutative fields $\overline{\mathbb{R}}$ and $\hat{\mathbb{C}}$ are self-irreducible, or holistic, within hypercomputation, with respective algebraic depth 1 and 2. This confirms the fundamental role that $\mathbb{R}$, $\mathbb{C}$, and ∞ play in mathematics.

CC: The computational world you describe, while fascinating, is about numerical computation, hardly suited to be the only model of *imago mundi*. Non-numerical computation is as important if not more important than numerical computation.

FC: I must confess that I do not find your distinction numerical/ non-numerical as being always helpful. The age-old mathematical tradition tends to accept as

"numbers" all entities over which one can perform computations of some sort in a meaningful way. Nature has more imagination than mathematicians and computer scientists put together... And Nature's computation is a boundless domain, which extends far beyond what you call numerical computation (which, I suppose, refers to real scalars only).
As open-ended as hypercomputation may be, it would be risky to claim that such a computational mode is the only model for the inner ways of the human mind. At best, such a model does capture some important features which are not revealed by classical computation. It is clear to me that most features of the mind remain outside the reach of hypercomputation.

CC: Please explain the difference between thinking in $\mathbb{R}$ (reals) vs. $\mathbb{C}$ (complex numbers). Is this related to 1D–thinking vs. 2D–thinking?

FC: Yes, the two expressions are equivalent, where $\mathbb{R}$ or 1D and $\mathbb{C}$ or 2D equally refer to the extended line $\overline{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$ and to the extended plane $\hat{\mathbb{C}} = \mathbb{C} \cup \{\infty\}$. Observe that $\overline{\mathbb{R}}$ (resp. $\hat{\mathbb{C}}$) is isomorphic to a circle (resp. Riemann sphere). Each set is associated with a different causality which shapes the corresponding human thinking. For example, a realisation of the causality based on $\overline{\mathbb{R}}$ is the scientific causality where cause always precedes effect. This is the causality that shapes the mind of an infant: the spoon invariably drops on the floor when not held in hand. Examples of the causality based on $\hat{\mathbb{C}}$ are not experimentally obvious. We all know that Western science rules out any causality other than the one—originally suggested by empiricism—which is based on $\overline{\mathbb{R}}$ (interpreted as the physical time line). However it is not difficult to pinpoint examples of $\hat{\mathbb{C}}$-causality in the act of mathematical understanding at once, in a flash of light. This is well documented in the writings of Poincaré (1908) and Hadamard (1945). In mathematics, understanding and discovery are global processes based on $\hat{\mathbb{C}}$, whereas proofs and communication are sequential processes akin to language: they are based on $\overline{\mathbb{R}}$ endowed with a natural order.

CC: Your views—from the perspective of continuous mathematics—on understanding, discovering, proving and communicating are very interesting. I would argue against the generalisation you made when you said "[i]n mathematics": in discrete mathematics, an increasing part of "mathematics", $\hat{\mathbb{C}}$ and $\overline{\mathbb{R}}$ play no role and understanding, discovering, proving and communicating appear in a different light.

FC: Of course there are discrete versions of $\mathbb{R}$- and $\mathbb{C}$- thinking adapted to hypercomputations realised on finite rings. Circles and spheres have to be replaced by regular polygons and polyhedra. The smaller the number of vertices, the greater the difference between certain aspects of discrete and continuous hypercomputation. Discrete thinking sheds a new light on the algorithmic role of old notions

such as the Sierpinski's triangle (i.e. the arithmetic triangle mod 2). A good part of my book is devoted to the constructive interaction between the discrete and the continuous in computation (Chapters 6, 8, 10). A remarkable example of such a constructive interaction is provided by the Picard iteration for the fixed-point equation $x = rx(1-x)$, with $-1/2 \leq x \leq 3/2$ and $-2 \leq r \leq 4$. The amazing properties of the logistic iteration: $0 < x_0 < 1, x_{n+1} = rx_n(1-x_n), n \geq 0$ were first reported in the West by the biologist R. May (1976). The descriptive complexity is low, but the global dynamics is complex (not real) in relation with the roots of $w^3 = -1$ and $z^6 = 1$ (Chapter 6). A complete analysis of this dynamics requires the cooperation of results by Fermat, Euler, Riemann, Cantor, Sierpinski, Sharkovski and Feigenbaum. These results run over a period of about three and half centuries. They deal with numbers in $\mathbb{N}, \mathbb{R}, \mathbb{C}$ and with actual infinity. The dynamical analysis blurs the three distinctions: discrete/continuous, real/complex, potential/actual infinity. These distinctions play an essential role in our current views about numerical computation. But they are intimately intermingled by the logistic iteration and defy separation. More is happening. At the three exceptional values $r = 1$ (resp. $-2, 4$), the iterate x_n can be expressed in closed form as a function of x_0, n and r by means of an exponential (resp. trigonometric) function. The algebraic and transcendental viewpoints coalesce at the three values $r \in \{-2, 1, 4\}$.

CC: The discussion of the Borel-Newcomb paradox in your book is fascinating. Can you please summarise it?

FC: I am pleased that you ask this question because the Borel-Newcomb paradox, which lies deep at the heart of nonlinear computation, has not yet received the serious attention it deserves.

In 1881, the American astronomer S. Newcomb noticed with amazement that the first decimal digit in numbers arbitrarily chosen among those produced by human computers or by Nature was about 6.5 times more likely to be 1 than 9. A paradox (Chatelin, 1996) emerges when one contrasts Newcomb's little known remark with the much better known result by Borel (1909) from which follows that all (but the first being $\neq 0$) decimal digits of a real number x chosen at random are uniformly distributed with density $1/10$ (i.e. x is simply normal in base 10). The paradox can be easily resolved by the theory of P. Lévy (1939): Borel and Newcomb do not look at the same numbers. The reals of Borel are abstract mathematical numbers free from any computational process. By contrast, Newcomb observes numbers which result from various *nonlinear computations* performed by man or Nature alike. And Lévy proved that computation makes the first digit more likely to be 1. This is a fundamental, yet little known, consequence of the scientific notation for a positive real x in the base $b \geq 2$: $x = sb^{\nu}$, where $s \in [1/b, 1[$ is the significand and $\nu \in \mathbb{Z}$ is the exponent. Observe that $x = b^{\log_b x} = b^{[\log_b x] + \{\log_b x\}}$, so that $s = \frac{1}{b} b^{\{\log_b x\}}$, where $\{\log_b x\} \in [0, 1[$ is the mantissa of x. According to Borel

(resp. Lévy) $\{x\}$ (resp. $\{\log_b x\}$) is uniformly distributed on $[0, 1]$.
Scientific computers use a version of the scientific notation adapted to the finite world (finite number of digits for the significand, finite range for the exponent): this is known as the floating-point representation of machine numbers. It is quite remarkable that floating-point computation transforms Borel-normality ($\{x\}$ uniform) into Lévy-normality ($\{\log_b x\}$ uniform). This indicates that the bad reputation of the inexact computer arithmetic is largely undeserved. On the negative side, the arithmetic is *not* exact; but the inaccuracy can be kept under control by a careful use of reliable numerical software. On the positive side, this floating-point arithmetic endows a piece of hardware with an epistemological value. It is perplexing to realise that this amazing potential is either ignored or dismissed by most scientists. Finally, let me add that the complete theory of Lévy includes the case of scalars in any finite ring $\mathbb{Z}_n, n \geq 2$.

CC: Do you subscribe to the view according to which "mathematicians think and reason, do not compute?"

FC: Literally speaking, such a view may describe some pure mathematicians of the twentieth century only. From Antiquity to the nineteenth century, all mathematicians did compute. Fermat, Euler and Riemann are three famous examples of computing mathematicians in the 17th, 18th and 19th centuries. But this is looking at one side of the coin only. Since the 16th century and Viète's idea to use letters to denote arbitrary data or unknowns, algebra grew increasingly symbolic, showing that reason is partly an abstract computation. This complementary side of the nature of mathematics is illustrated by Bourbaki: maths is presented as an all-axiomatic, abstract, most general formal construction based on set theory. Both aspects are equally at work in mathematics, but they are unevenly distributed among mathematicians.

CC: The twentieth century "paradigm shift" in computing appeared when, to cite from your book, the "notion of *mathematical computability*, crafted over millennia, was abandoned in favour of *mechanical* computability ..." What is the "millennia old" *mathematical computability*?

FC: The red thread which runs through the historical development of mathematics has been the crucial question of which entities could be accepted as *numbers*, so that one could confidently compute over them and get *meaningful* results. This epistemological question is "mathematical computability." Before they could be accepted as *bona fide* numbers, the status of the following entities has been passionately scrutinised: irrational numbers, zero and $\infty = \frac{1}{0}$, negative numbers, complex numbers, quaternions, to name a few familiar examples. The list of numbers is open-ended; it contains increasingly symbolic "numbers", most of which are yet to be thought of.

CC: We talk about numbers—old, new, or yet to be discovered, but computing is not about numbers only. Computability theory, developed in the twentieth century, is a mathematical theory of mechanical computing. It shows that one can compute non only with numbers (of any type), but also with strings over alphabets. The digital revolution has its main roots in the "computer on paper" imagined and studied in this field. The universal Turing machine is the theoretical model justifying the very existence of our current computers, from laptops to the super-computers. Gödel's incompleteness theorem (and many other results which followed) was proved in this framework. Still, you seem to hold strong negative feelings against this field...

FC: First, let me slightly disagree with you. Why should letters in an alphabet be considered of a different nature than numbers, if they enable well-defined computations? They are both symbols conceived by man. In the past century, Poincaré and Hilbert pioneered the use of group to characterise all isometric transformation in geometry. Should we call these "numbers" or "letters"? Either way is fine of course. Under the influence of cybernetics and molecular biology, the more encompassing but vaguer term "information" seems to gain interdisciplinary acceptance.

The sweeping success of computers in our society owes something to the theory of computability. Arguably, it owes also to the floating-point arithmetic of scientific computers. In addition to 1936 Turing's paper, in my view, the origin of the computer revolution could be assigned to the year 1914 which saw the first known design of a floating-point computer arithmetic, by L. Torres y Quevedo in Madrid. According to Knuth, the scientific notation was used implicitly in Sumer four millennia ago! Moreover, if scientific computers were to disappear tomorrow, this would deal a severe blow to the development of the scientific know-how at the two extreme scales, large and small, such as cosmology and high energy physics.

Second, I want to protest. I certainly hold no negative feelings against the respectable field of computability theory. It is very important to delineate the precise limits of mechanical computing, the limits of formal axiomatic systems. My reservations only concern the claims of some researchers which extend to mathematics, without further ado, results which have only been proved for Turing machines; they ignore the warning of Gödel himself against such a bold extrapolation (1972).

CC: With your permission I wish to continue our disagreement. Floating-point computer arithmetic is part of numerical computing which has many applications in science and technology. However, we should not forget the equally (maybe more?) important area of non-numerical computation. The use of internet, social networking and many other non-numerical computations—the largest part of computer science daily used by laymen—are modelled by Turing computations,

not by a computation in a commutative or non-commutative algebra.

FC: What about Boolean algebras? And is not $\mathbb{Z}_2 = \{0, 1\}$ a commutative field? Your question is a good illustration of the current cultural divide between mathematics and computer science. A divide which expresses only a difference in perspective about computation. Turing's original diagonal argument (1936) tells us that the algebraic structure of $\mathbb{N} \times \mathbb{N}$ can be modified by computation: unbeknownst to the user, algebra creeps in...
Now, the opinion that computer science is more (or less) important than scientific computing is also a matter of perspective. Computers have certainly a visible impact on the organisation of everyone's life. The contribution of computer simulations in engineering is not obvious on a daily basis, but the results are highly visible in high tech industries, from cellular phones to nanotechnologies. However, beyond academia, computers are not used as strict Turing machines. They harmoniously combine, whenever needed, the two aspects of computation which are logical and numerical. It is useful to contrast—but useless to oppose—these two kinds of computers.

CC: I certainly agree with your last point of view. But language itself can contribute to the mutual misunderstanding. For example, the name "scientific computing" seems unfortunate: it creates the (wrong) impression that any other type of computation is unscientific. Some terms like computability, complexity, hypercomputation, have different meanings in computability theory and in "classical mathematics." For example, the meaning of hypercomputation in the sense you cited is different (and older) than the one practiced in computer science (which means "going beyond the computational capability of any Turing machine").

FC: This results from natural evolution: new fields of knowledge tend to drift away from their origin as they mature. This happened to computability theory which in 80 years evolved away from mathematics to get closer to computer science. Inevitably, this creates a semantic ambiguity which should be clarified when it arises. For example, what is mechanically computable differs greatly from what is mathematically computable.

CC: You write that "The coup de force was accomplished in the name of *rigour*, neglecting the fact that rich polysemic notions are necessarily ambiguous. Only trivialized notions can be crystal clear." Rigour doesn't oppose ambiguity (think about fuzzy sets theory or polyvalent logics).

FC: You are absolutely correct about alternative logics. The example of fuzzy logic is developed to some extent in my book (Chapter 10). My first comment is to recall that fuzzy logic (Zadeh,1961) is extremely *controversial* in the West, despite its impressive successes for smart technology. This is not surprising since I show that it is related to $\hat{\mathbb{C}}$-causality, a blind spot in current science. My second

comment is that, when the coup de force took place in the 1930s, it was explicitly directed against geometry. It had been realised by then that geometric evidence was not fully rigourous. But the mathematicians threw away the baby with the bathwater, and $\hat{\mathbb{C}}$-causality was implicitly banned together with geometry. This explains the paramount importance given afterwards to axiomatisation and formal proofs in any mathematical work intended for publication (papers and textbooks). This formal approach—duly avoiding any reference to meaning, interpretation, not to mention significance for life—has deterred many a scientist and schoolchild from mathematics.

CC: Citing again from your book: "Turing's proof of the unsolvability of the halting problem is a mechanical version of Gödel's incompleteness result: because $\mathbb{Z}$ is algebraically too "poor", there will always exist propositions written in a formal axiomatic system which happen to be true but are not formally provable within the system." Unsolvability and incomputability abounds in continuous mathematics as well. Just to quote a well-known example: Pour-El and Richards proved in 1979 that an innocent looking differential equation has uncountably many solutions, but does not have a computable solution.

FC: I stress in my book that the deep reason behind Gödel (1931) and Turing (1936) is the fact that the set of numbers that they consider has an algebraic depth equal to 1 rather than being ≥ 2. The algebraic depth is related to the purely algebraic notion of a derivation map. To recall, a derivation D over an arbitrary multiplicative algebra A is a linear map which satisfies, for any $x, y \in A$ the Leibniz formula: $D(x \times y) = (Dx) \times y + x \times (Dy)$. This extends the 17th century old notion of derivative for a function to a much broader context, which can be either discrete or continuous. Thus the question of discreteness ($\mathbb{N}$ or $\mathbb{Z}$) versus continuity ($\mathbb{R}$) is *not* essential. For example, Rumely (1986) showed that Hilbert's 10th problem is solvable over algebraic integers which have an algebraic depth equal to 2. Therefore the result of Pour-El and Richards is not surprising. However, in a continuous context, it is important to avoid the semantic ambiguity. A precaution not taken by these authors.

CC: Maybe the result proved by Pour-El and Richards is not surprising today, but it belongs to a large list of undecidable problems in continuous mathematics, analysis, real or complex, topology, differential geometry, quantum physics, to name just a few. Apparently these results have nothing to do with algebraic depth.

FC: Pour-El and Richards result is a welcome reminder that there exist natural phenomena (modelled by continuous mathematics) which are mathematically computable, but cannot be formally produced by mechanical computation. Unless one's mind is fully under the spell of finite algorithms, this should not come as a big surprise. Admittedly, the many Turing-like undecidability results in con-

tinuous mathematics have been proved without any explicit appeal to the notion of algebraic depth. However, this does not mean that the notion should play no explanatory role. Indeed, Rumely's result points in the other direction. But to perceive such an implicit role, it is useful to think outside the Turing box, and apply the wisdom credited to Hadamard: to fathom the reals, go complex. One might want to expand the framework of Turing computability to the broader one of hypercomputation, while staying within the scope of mathematical computability.

CC: Can you give your arguments refuting Church-Turing thesis?

FC: This is certainly not an absolute refutation. Our dialogue has shown the importance of the algebraic *context* for any computability theory: binary vs. real, rational vs. irrational, algebraic vs. transcendent, real vs. complex and so on ... What I question is the claim to absolute universality of the Church-Turing thesis. If universality is meant *within today's machines* using classical technology, then I agree that the thesis is plausible. But if universality refers to mathematical computation, then the thesis is obviously too restrictive. It cannot account for the simple fact that $\sum_{k=0}^{\infty} 2^{-k} = 2$. The real number 1.1111.... in base 2 is Turing-computable but cannot be identified with the integer 2 by any finite algorithm. This inability prevents the Turing machine to achieve life's simplexity on its own. In addition, it is telling to contrast Turing machines and scientific computers. It is known that Borel-normal sequences appear random to any finite-state machine. Analogously, any Chaitin-normal sequence (i.e. algorithmically irreducible in computer science parlance) appears random to any Turing machine. From an epistemological point of view it does not pay to move from a finite-state to a Turing machine. You can only get an *aggravated* form of randomness (Chaitin, 1977). This makes a sharp contrast with scientific computers!

CC: A tuatara machine (Calude, Stay, 2006) has the capability you seem to think a Turing machine may miss.
Why do you think that, epistemologically, you don't need Turing machines, you can do everything with (the weaker) finite-state automata? Also, please explain the "sharp contrast with scientific computers!".

FC: This is not at all what I think. First, I do not say that the epistemological alternative to a Turing machine is a finite-state automaton. I say that the alternative is a scientific computer. The contrast between the two computers appears in full light when one starts from a Borel-normal sequence which looks random to a finite-state automaton. By processing the sequence on a Turing machine, one is likely to get a Chaitin-normal sequence. Such an algorithmically irreducible sequence is tautologic: each of its bits has a value justified by no other formal reason than itself. In other words, the sequence is because it is; there is no deeper reason to be found. For x chosen at random in [0,1[, the global identity $x \mapsto x$

becomes a string of tautologies at the bit-level inside x. Now, if we process a Borel-normal sequence on a computer endowed with scientific notation, one is likely to get a Lévy-normal sequence. The dynamics differs because the map $x \mapsto \{x\}$ is replaced by $x \mapsto \{\log_b x\}$. Instead of the reinforcement of the identity that is produced by a Turing machine, the scientific computer induces a change of viewpoint, from x to its mantissa. Interestingly, “mantissa" in Latin means “additional weight," whereas its Greek root “manteia" means “divination".
A Turing machine is the tool of choice to analyse identity and invariance; it enforces the chosen law of logic. A scientific computer, on the other hand, is a tool designed to model natural change and evolution, thanks to its floating-point arithmetic. The goal is no more to maintain the logical coherence, but rather to explore the evolutive diversity of Nature. In conclusion, neither of these two versions of computers can claim supremacy over the other. Both provide an invaluable help in the scientific approach through computation of Life’s phenomena, which combine constructively invariance AND evolution in amazingly inventive ways.

CC: Many thanks!