trópos – ISBN 979-12-218-0119-4
DOI 10.53136/97912218011945 pp. 59-75 (maggio 2021)

UNDERSTANDING LIVING BEINGS BY ANALOGY WITH COMPUTERS OR UNDERSTANDING COMPUTERS AS AN EMANATION OF THE LIVING

MAËL MONTÉVIL(*)

Abstract: The analogy between living beings and computers was introduced with circumspection by Schrödinger and has been widely propagated since, rarely with a precise technical meaning. Critics of this perspective are numerous. We emphasize that this perspective is mobilized to justify what may be called a regressive reductionism by comparison with physics or the Cartesian method.
Other views on the living are possible, and we focus on an epistemological and theoretical framework where historicity is central, and the regularities susceptible to mathematization are constraints whose existence is fundamentally precarious and historically contingent.
We then propose to reinterpret the computer, no longer as a Turing machine but as constituted by constraints. This move allows us to understand that computation in the sense of Church–Turing is only a part of the theoretical determination of what actually happens in a computer when considering them in their larger theoretical context where historicity is also central.

Keywords: Theoretical Computer Sciences, Theoretical Biology, Exosomatisation, Constraints, Calculability, Epistemology.

(*) mael.montevil@gmail.com.

1. Introduction

Theoretical computer science comes primarily from the debates in mathematical logic at the beginning of the 20th century. With the appearance of contradictions in mathematics in the late 19th century, mathematicians and philosophers turned to logic, and the formalization of mathematical proof, to ground the latter on a reliable basis.

However, Gödel's theorems challenged this project by showing the intrinsic limits of logical formalisms. Gödel's theorems simultaneously introduced the notions of coding and the incalculable. The latter is, thus, in a sense, an origin of computer science and computers. Indeed, the incompleteness demonstrated by Gödel means that some assertions, formulable in a logical theory rich enough to accommodate arithmetic, are neither provable nor refutable in the same theory. Theoretical computer science is thus more precise and subtle than some contemporary regressive rhetoric asserting that everything is computable (Anderson 2008). These discourses nevertheless focus on a slightly different question than the one at the origin of computer science. Gödel's theorems are entirely within mathematical logic, whereas these discourses are primarily about the relationship between computer science and the natural and social sciences. We will come back to this point.

One of the consequences of the work in mathematical logic is the design of the computer, notably via the work of Turing. Turing indeed introduced a logical formalism based on the schema of a machine that reads and writes on a tape, according to finite, automatic rules. This formal machine is equivalent to other logical formalisms defining computation: Church–Turing thesis posits the equivalence of the various formalisms defining what is computable. Note that this thesis has a rare epistemological status in the field of mathematics. It cannot be proven because there is no formal definition of all possible formalisms. We need, for example, two formalisms to prove that they are equivalent and obtain a theorem that is limited to these two formalisms.

Computers thus come from a logical–mathematical question: what can we deduce from axioms, or, in terms of Turing machines, what computational processes terminate? We insist on this central point: these mathematical frameworks enable us to understand what the machine can do. The rule of the computation performed by the machine and its inputs are given by hypothesis. Thus, from the logics' perspective, it has the status of an axiom.

Theoretical computer sciences do not account for the role of programmers and users that generate computers, their programs, and input. As such, it has a limited scope. In this context, Bernard Stiegler argued for the need for a new perspective for theoretical computer sciences, embracing its position in an exorganological framework in a discontinuous continuity with biology. Then computers emanate from human activities and, at the same time, transform them.

When computers were designed, biology was investigating the nature of heredity, that is to say, the resemblance between subsequent generations and, most importantly, the appearance of heritable variations that are the basis of the Darwinian framework. The physicist Erwin Schrödinger speculated that the material support of heredity was some aperiodic crystal, thus a discreet structure (Schrödinger 1944). Building on this idea, he proposed that the notion of "code script" could be the basis of heredity, thus introducing the analogy between the new emerging machines and the living. In this sense, evolution would provide the programs, and organisms would unfold them. As a seasoned theoretical physicist, Schrödinger draws the consequences of this assumption, notably the laplacian structure of determination that follows, that is, the deterministic and predictable nature of the processes hypothesized. After the subsequent discovery of DNA and notably the work of Jacob and Monod (Monod 1970), the analogy between living beings and computers shaped biological sciences, especially molecular biology and, more broadly, experimental biology.

What are then the relationships between the concepts of theoretical computer science and the living? Should the living be understood with these concepts, should the two fields be sharply separated, or could biological concepts renew the perspective on computers?

This article will first come back to the transfer of computer science concepts to understand living beings and criticize them. Then we will introduce some recent concepts and theoretical frameworks from biology, moving away from the epistemology of both physics and computer science in favor of a new articulation between natural phenomena and mathematics — we emphasize the central question of historicity in that regard. Finally, based on these concepts, but without reducing the question to them, we will reconsider the theoretical perspective on computers from an exorganological perspective in the sense of Stiegler.

2. The shortcomings of understanding biological organizations as genetic programs

As mentioned in the introduction, the use of the computer program paradigm in biology initially targeted heredity. It also built on the Weissmanian schema, namely the idea that the support of heredity determines the organism without being determined by it. The subsequent discovery of DNA led to the molecular biology revolution.

It is not the place here to discuss in depth the history and the general criticism of this conception (this has been done elsewere, e.g. Keller 1995; Longo *et al.* 2012; Longo and Mossio 2020; Walsh 2020; Soto and Sonnenschein 2020). Instead, we will focus on the practical outcome of this paradigm, chiefly the molecular biology practice — this practice couples two distinct dimensions concerning the computer analogy.

First, this analogy led to precise empirical practices. Biologists investigate the interaction between molecules around the DNA (such as RNA, proteins), often correlating them experimentally — and only experimentally — to very macroscopic aspects, such as shapes, processes, or behaviors. Dramatic examples are the putative genes of homosexuality or intelligence. In this practice, the passage from the molecular to the macroscopic level is never made explicit theoretically. A kind of deus ex machina is required to articulate the two levels. Incidentally, this shortcoming is reminiscent of the epistemological concerns with vitalism beyond the metaphysical ones. The vital forces were sometimes described by analogy with classical physics forces; however, they were not made explicit; they were also ad hoc explanations.

Second, to fill this vacuum or provide the illusion to do so, biologists build on a vague discourse. In a nutshell, the DNA contains the information for the development and physiology of the phenotype; biologically relevant processes follow a program that should be the object of biological investigation. Therefore, biologists assert that they are elucidating the underlying program by examining interactions between molecules originating from DNA. In a sense, DNA becomes a kind of first immobile engine; any explanation of relevant biological phenomena must go back to it. In practice, this approach is justified by the hope to find magic bullets, allowing to cure or transform the living according to a specific end.

Biologists never work on the putative program synthetically; it is only investigated locally by small manipulations, or recently, a little more broadly

by high–throughput methods. Nevertheless, the gap with the phenotypes remains complete.

This perspective goes with a mechanical view of the living to fit the Laplacian nature of the Turing machine — against the view of Turing himself (Turing 1950, 1952). This view and the Weissmanian schema conflict with physics developments since Newton, where reciprocal causality is a principle. Moreover, physics developed a rich perspective where mathematics is central, provides the backbone of theories, and brings forth absolute limits to the possible hubris of scientism, for example, against the possibility of perpetual motion or the ability to predict every phenomenon.

By contrast, the mathematical developments of molecular biology are limited to statistical inference. Causal mathematical schemes are mostly limited to linear causation. This situation is paradoxical because, for biologists, molecular biology is a physicalization of biology. It is not entirely false since molecular biology understands the substrate of heredity in physics terms; however, its understanding of phenotypes remains very limited.

Let us emphasize that this methodology and perspective is not a reductionism in the usual sense of the word. For example, the Cartesian schema requires decomposing an object to understand its parts and then to recompose it, at least theoretically. Molecular biology only performs the decomposition step. The postulated primacy of DNA ascertains that the decomposition is relevant, but there is no theoretical recomposition. In practice, the observation of the phenotype replaces observations. Accordingly, this perspective does not fit the physics picture of reductionism. In physics, reductionism means describing a system at the microscopic level of description; by contrast, in molecular biology, it means looking only at specific microscopic parts of the system. Molecular biology considers itself a physicalism, but it is more reductionist than physics itself. In this context, mainstream molecular biology does not work on its theoretical constructs. Another illustration is the system concept that is ubiquitous in physics and mostly limited to the minor subfield of systems biology in the study of organisms. To conclude, this reductionism is more a regressive tendency than a genuine reductionism.

3. Constraints and historicity in theoretical biology

In this section, we will introduce some recent concepts of theoretical biology that provide an alternative to the computer metaphors and emphasize the

question of the role of mathematics in the field, a question that is also relevant for computer science.

The starting point of these concepts is the theorization of the historicity of the living, obviously in line with the theory of evolution. Unlike population genetics, our focus is not the mathematization of specific evolutionary mechanisms. Instead, we focus on the theoretical and epistemological counterpart of this historical character of life, particularly concerning its mathematization.

Mathematization in the natural sciences comes historically from physics, and it is the most refined in this field. The situation may be described concisely by stating that this mathematization is based on the idea that the changes taking place in a phenomenon can be understood by an invariance more fundamental than these changes (Bailly and Longo 2011). Accordingly, the space of possibilities is pre–given. The trajectories or final structures of a phenomenon derive from predefined structures, such as the symmetries given by theoretical principles; for example, the space–time symmetries of Galilean, special or general relativities.

If we take the historicity of life phenomena seriously, we can no longer understand change by underlying invariance. On the opposite, it is not only necessary to understand how the forms of living beings change. We also need to understand the changes in their physiologies, their modes of reproduction, their functions, the structure of their heredity, and the invariants that we sometimes seem to be able to discover by looking at certain specimens. It is then a question of postulating historicity, and notably the fact that living beings can vary in a strong sense, that is to say without underlying invariance (Montévil *et al.* 2016). Once this point is taken, we do not need to abandon the concept of invariance in biology, but specific invariance no longer stems from first principles. A given invariance is then local, limited to a more or less broad class of living beings, and contingent in the sense that some specimen may escape it. We have called these local invariants constraints (Montévil and Mossio 2015; Soto *et al.* 2016).

Constraints have several theoretical roles. First, they canalize and structure transformation processes. For example, DNA canalizes the processes of protein production. Similarly, the bones of the arm limit the possible movements of the latter. In doing so, the constraints enable processes that would not take place without the constraints. For example, without DNA, randomly formed proteins would rarely be functional, and without the bones of the arm, most of its movements would not be possible. Constraints also limit, among other things, the default state of cells: proliferation and motility.

Second, another remarkable property of constraints in an organism is that they maintain themselves collectively via constrained processes (Montévil and Mossio 2015). Thus, DNA sequences constrain the transcription of messenger RNA, which constrains the production of proteins, and, among the latter, some constrain various processes that maintain the structure of DNA. The same type of circularity is found at much more macroscopic levels, for example, between vertebrates' organs. This property, called the closure of constraints, does not mean that the organism maintains itself identically. It just maintains its constraints so that they last against the spontaneous growth of their entropy and disappearance of their invariance (which remains local and, in a sense, contingent since they have a historical origin).

Finally, constraints play a diachronic causal role in the sense that they allow the appearance of novelties (Montévil *et al.* 2016). Thus, for example, articulated jaws have allowed the appearance of teeth and all sorts of functions such as the protection of eggs in certain ray–finned fish (for example *Opistognathus aurifrons*), the transport of youngs, or articulated speech in *Homo sapiens*.

If the concept of constraints addresses certain aspects of a given organism, it does not entirely define this organism in a given context. In physics, the theoretical definition of an object stems from its invariants and symmetries, and more generally, by a mathematized theoretical framework. It follows that the theoretical object is generic so that all electrons, for example, follow the same equations — they have no singularity in the philosophical sense. This point has very practical consequences. For example, the light speed in the vacuum defines the meter. It is an invariant introduced by Einstein and corresponds to the speed of any light ray in a vacuum (or any photon from a corpuscular perspective). The ability to define objects theoretically in this way also allows some separation between the concrete object and the theoretical object: it is unnecessary to anchor the theoretical object to a specific concrete object. For example, the standard meter is only a practical device; if destroyed, physicists could construct a new one with the same length with very high precision. In biology, there are no such theoretical constructs because, on the one hand, organisms embed a multiplicity of particular constraints that appeared over time, and, on the other hand, these constraints continue to change, even under standardized laboratory conditions (Montévil 2019).

It is then interesting to recall the remarkable epistemological originality of the phylogenetic method of classification of living organisms (Lecointre and Le Guyader 2006). This method does not aim to describe the living beings by

the relations between their parts and invariance in these relations, as in physics. Instead, biologists define classes as the set of descendants of a common ancestor. The latter is theoretical; biologists do not identify it concretely. In practice, specimens are therefore analyzed by their estimated similarities, provided that the latter is evidence of their relatedness. For instance, mammals have shared characters that birds do not have. Therefore they most likely have a common ancestor that birds do not have and form a class. In a sense, since biological objects cannot be described by invariants derived from their theoretical determination, biologists rely on another type of invariance: the shared past of these objects, a past defined by the genealogy underlying the theory of evolution. Moreover, since biologists define the objects by their past, this theoretical framework can accommodate unpredictable and unprestatable variations.

This method has a second point which is significant for us. The operational definition of a class can neither be based on a common ancestor because the latter is unknown nor on the invariance of its causal structure because the latter varies. It, therefore, requires reference to a particular specimen, called a holotype. The holotype is not the common ancestor used to define a class but rather a reference point for fixing the meaning of a name in the classification. The name is then extended to the intended class, provided that the specimen is part of it. Contrary to physics, the reference to concrete objects is necessary for this classification's epistemology. By extension, it is necessary for biology because the classification gives the names of the objects it studies (at the organism level). Of course, in the experimental practice of biology, other elements can be added to the definition of a biological organism, such as the known genealogy, when it corresponds to animals raised in laboratories and the environment in which they live. However, these considerations do not change the epistemological move of defining objects largely by their past rather than by what they do.

Mathematical writing, in physics, is based on the invariance of relations between relevant quantities. However, this method does not correspond to the theoretical and epistemological conditions of biology. In order to build on the epistemology of historicity outlined above, a new type of symbol has been introduced in theoretical biology, noted χ (Montévil and Mossio 2020). This theoretical object is a symbol rather than a quantity or a variable, like in physics' formalisms, because this symbol's epistemology requires the reference to a concrete object, for example, a holotype in the phylogenetic classification. This general approach can be adjusted to the diversity of the situations encountered.

This symbol fulfills several epistemological roles. First of all, it enables us to account explicitly for definitions of objects by their past within their formal description — though the nature of χ is not formal in any usual sense. It also aims at accommodating, within these formalisms, the possibility of variations whose nature cannot be predicted; that is, the appearance of novelty in a strong sense. Again, this does not mean abandoning constraints as local invariance but formally articulating them to this kind of symbol. Then, for example, we can account for constraints whose primary function is to generate novelties whose nature is not pre–given. The articulation of χ and constraints allows the latter to be explicitly historicized. However, this framework implies that a possible variation can always break the validity of a precise constraint. The latter is more or less frequent and significant depending on the constraints and experimental conditions considered.

The symbol χ accounts for a part of the theoretical determination of the biological object that is not captured by an underlying invariance and therefore does not allow for computations, particularly concerning the appearance of functional novelties. The originality of this approach is the articulation of this incalculable with precise epistemological and methodological considerations that account for essential elements of biology: the classification of living organisms and numerous aspects of experimental practices. These considerations are frequently forgotten in other fields of biology, typically experimental biology, because of epistemological perspectives inherited from physics, without these fields meeting the theoretical conditions of these physical theories (the definition of explicit, general theoretical invariants and the subsequent mathematical definitions of theoretical objects). Here, the new symbol comes from the crossing of two epistemologies, the relational epistemology of mathematical modeling as practiced in physics, and manifested here through the concept of constraints and the historical epistemology coming firstly from evolutionary biology — it is also relevant for development.

4. Towards a new theoretical computer science

In order to progress on the question of the relationship between computer science and the living, we propose to reinterpret the object of theoretical computer science and then transfer some concepts of theoretical biology in this field while keeping a critical view on this transfer. In this sense, this work is a

contribution to answer the call of Bernard Stiegler to rethink the foundations of computer sciences (Stiegler 2020).

4.1. *Theoretical computer science as a human science*

The perspective of calculability and the Church–Turing thesis mentioned in the introduction has been an end for the design of computers, machines allowing to perform calculations in the sense of Church–Turing's thesis. The calculations carried out by a computer are then defined by programs by abstracting from the material, concrete realization of the machine. An essential part of classical theoretical computer science is the elaboration of a diversity of formalisms allowing to think differently about what a program is, but always with the idea that these formal changes do not transform what is computable (except for simple languages that do not allow to compute all the functions of Turing machines).

Conceiving theoretical computer science from the calculation of the isolated machine was justified at the origin of computers when the main difficulty consisted in making this new type of machine exist. It seemed all the more justified since the computer was posed as the mechanization of the part of the human mind that philosophers like Frege thought to be the most rational and the most reliable, that is, logic. Today computers are omnipresent, in various forms, and networked. In this context, this perspective seems to us quite insufficient because it does not consider the consequences of computers on noesis.

Our answer to Bernard Stiegler's call to rebase theoretical computer science consists of changing its theoretical object. Rather than considering the machine carrying out its calculation in isolation, considering the machines in connection with noetic beings, that is, thinking beings in the sense that thinking brings about the capacity to take care of a new situation.

This theoretical move has several immediate consequences. The first is that theoretical computer science should not limit itself to considering the effects of programmers and users on machines but should also consider the effects of machines on noetic beings. In particular, given that computers depend on human knowledge to exist and if computer science leads to an excessive proletarianization, that is to say to a loss of this knowledge, then it risks leading to the destruction of its own conditions of possibility. Theoretical informatics could then possess an internal coherence, but it would nevertheless be fundamentally irrational.

The second consequence of this move is that classical theoretical computer science only deals with a very particular case, where the machine is left alone to perform its computation. However, this is not what concrete computers typically do, they are used interactively, and programmers and users commonly transform the rules of their computations. It follows that classical theoretical computer science does not allow us to understand the actual (observable) trajectories of these objects that are computers. From this point of view, classical theoretical computer science would become a limit case of the new theoretical computer science in the same way that classical mechanics is a limit case of general relativity, the case where velocities and masses are small. In computer science, it would be the case where the input and programming of the machine are given, and the machine computes in isolation. In this sense, classical computer science is the limit case where its functional insolation sensu Dwivedi and Mohan (Dwivedi 2020) is considered absolute.

Admittedly, there are theoretical approaches to deal with parallelism, for example, when several users online are trying to order a single available concert seat. However, these approaches only ensure that the program always follows the purposes of the programmer, his employer, or client). In this case, the problem is to ensure that only one user can pay for this single available space — the problem is then equivalent to the technical issues of parallelism (several calculations carried out in parallel, in an asynchronous way, which introduces randomness just like the activities of the users acting in parallel), which go beyond the strict framework of Turing machines, provided the latter are deterministic(Longo, Palamidessi, and Thierry 2010). However, these approaches are very far from theorizing user activities and the consequence of computers on them.

On the other hand, a framework, or rather the convergence of two frameworks, can be considered a sketch of extended theoretical computer science — a biased and particularly pathogenic sketch. As taught at Stanford, it is the convergence between computer and cognitive sciences, which is the basis of many platforms and video game mechanics designed to make the user addict, and more generally, to make his behavior follow the publisher's interests. This convergence does not think about biological and psychological development, although it aims at education in some cases, and it does not address the question of noesis, thinking, beyond some simple properties. However, it does emphasize the importance of considering a theoretical alternative to this convergence, an alternative that is not oriented towards a short–term economic opportunity but the care for computers and noesis.

4.2. *Insights from theoretical biology*

We would now like to suggest that some concepts of theoretical biology could be mobilized to give new perspectives to theoretical computer science. In the previous discussion, we emphasized questions of theoretical biology that we also think are relevant to rethinking computer science. However, this type of discussion always requires a critical distance. Some concepts, such as the default state of cells, have no obvious counterpart in computer science. On the other hand, noesis is not a biological concept.

Nevertheless, some biological concepts reflect on the articulation between historicity and mathematics, and their relevance is somewhat straightforward provided that we acknowledge the different theoretical contexts. In a sense, historicity is specific to the living, but the study of the living is not limited to biology. Human beings, human societies, and the artifacts they produce also participate in the living, with theoretical particularities, notably noesis. Bernard Stiegler worked on artifacts such as pen and paper, or computers, with the concept of exomatisation introduced by Lotka (Lotka 1945), that is to say, the idea that these objects are similar to organs constructed outside biological bodies. From this perspective, there is continuity between artifacts and parts of organisms.

Along this line, the first concept that we think relevant is the concept of constraint. A constraint is, first of all, a local invariance maintained far from maximum entropy. Thus, a computer or smartphone hardware is a constraint on the dissipation of free energy in electrical form, coming from the mains or a battery. They are also a constraint for programmers and users because they do not change and have a causal role in these processes. Let us add that, still, at the hardware level, the concept of closure between constraints is relevant. The hardware must be maintained, whether by upkeep, often limited to removing the dust that accumulates in the ventilation of a computer, or more often by replacing components or entire machines — the latter undergo entropy growth, which leads to their malfunction. The components where these phenomena are most noticeable are batteries, whose capacity decreases over time, and storage devices, such as hard disks and SSDs, which work thanks to a certain redundancy, such as the use of extra sectors to replace defective ones. Hardware also has a diachronic role because it contributes to making possible the appearance of new constraints, including the production of new hardware (today, it takes computers to build computers).

The concept of constraint is also relevant to understand software. Software canalizes, in particular, the user's activity, but it does not determine it as strongly as physical principles determine the behavior of its objects. As in the case of hardware, software code is actively maintained, notably by copying processes that allow it to last beyond the life of its medium. What programmers call software maintenance is, however, distinct. It means ensuring that software still works after updating the software it depends on and fixing security flaws that may be detected. Thinking of software as a constraint means that, just as the geometry and rigidity of the bones in an arm both constrain and enable its movement, software constrains what is possible while enabling or facilitating specific processes. In biology, some constraints have primarily the function of maintaining other constraints, while others, called propulsive constraints, have a more fundamentally diachronic role, participating in the appearance of new constraints. Let us note that, transposed into this vocabulary, tertiary retentions, like writing, are themselves also constraints.

At this point, it is interesting to compare the concepts of constraint and pharmakon. These concepts do not precisely cover the same issues, the concept of constraint being more local — it does not by itself include the question of the role of these constraints in an organization. Nevertheless, constraints have the ambivalence of the pharmakon in the sense that a constraint limits possibilities while constituting them. In this case, the question of the opening or closing of possibilities concerning software is eminently pharmacological… and a pressing question. The articulation between computer science and cognitive science mentioned above serves to tune the user's behavior to the editor's interests. In that case, it is typically by strategies based on a pathological addiction, where the capacity of users to produce new possibilities is strongly degraded. These questions naturally lead back to design and the question of the ends of design that should now be embedded in theoretical computer sciences.

Let us note that, on the side of programming, classical theoretical computer science is only concerned with the functioning of programs, thus leaving aside the changes of these programs, i.e., the programming processes. In a sense, these changes are paradoxically one of their central concerns. With the Church–Turing thesis, we have seen that all formalisms are considered equivalent in terms of what they allow to compute. In concrete practice, this means that all sufficiently rich languages allow computing the same functions. Why, then, introduce new formalisms and programming languages? The main reason, in our opinion, is that these different approaches to computation allow

treating problems from different perspectives and that some problems are easier to approach from one perspective or another. Moreover, in practice, languages can have a higher or lower level of abstraction from what happens in the concrete hardware architecture, abstraction having advantages like ease and portability to different architectures, and disadvantages like less precise process control and generally lower speed. Thinking of programming languages themselves, and more specifically their implementation in code–interpreting software such as compilers, as constraints helps to overcome this paradox. They act as a constraint both on the compilation or execution of code and programmers' activity.

This analysis is also relevant for the code itself, which acts as a constraint on two distinct processes. The code defines software as a constraint, and at the same time, it plays the role of a peer–readable text. This dual role manifests itself in particular through the comments, which have no role in the execution of the code but serve to facilitate its understanding. If this understanding sometimes has a pedagogical role, it also aims to allow the code to be reworked and changed. The comments thus play a diachronic role; in other words, they constitute propulsive constraints. In the same way, this dual role appears for the code used by the machine in the frequent compromise between optimization and readability. Let us quote Donald Knuth on this question:

Programmers waste enormous amounts of time thinking about, or worrying about, the speed of noncritical parts of their programs, and these attempts at efficiency actually have a strong negative impact when debugging and maintenance are considered. We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil. Yet we should not pass up our opportunities in that critical 3% (Knuth 1974).

Thinking about computer science with the concept of constraint also aims to rethink the link between computer science and mathematics. Classical theoretical computer science emanates from mathematics, and the mathematics used is essentially discrete. These mathematics correspond to situations where the measure can be perfect in principle, and the determination is Laplacian, as Turing himself underlines (Turing 1950). On the other hand, we have introduced the concept of constraint precisely to account for the limits of the mathematical description of biological objects. These limits stem from the collision of two distinct epistemologies, the one of historicity and the relational one. Introducing this concept in computer science means that computer science's theoretical object does not follow a stable mathematical framework

but has fundamental historicity. If we consider a given computer, the trajectory followed is no longer the unfolding of a program on a given input but a permanent relation between exosomatic constraints (hardware, software) and the user. A fortiori, this point of view is essential when the user changes the code of the software he uses — or, in a rarer but essential way, when he participates in the design and construction of hardware.

Bernard Stiegler often referred to Paul Claudel's sentence: «There must be in the poem a number such that it prevents counting» (Petit 2019). In computer science, to accommodate the incalculable without abandoning calculation, he defended the idea of introducing incalculable fields into this domain, notably to (re)give a role to deliberation (Stiegler *et al.* 2020). This perspective and the discussion on constraints lead us to consider introducing in theoretical computer science something like the χ symbol introduced in biology. We do not yet have an elaborate framework for this, but we can introduce some remarks. Here, the contribution of the phylogenetic classification of living beings is no longer really relevant, but the definition of the user by his history can be — thus joining medicine where the history of the patient is essential. The theoretical manipulation of χ depends on the issues at stake. For example, χ makes it possible to convey that knowledge is never purely synchronic; it is primarily diachronic. They are above all carried by singular persons and groups, which is epistemologically similar to the use of types in biology classification.

To conclude, theoretical computer science can be seen from two angles which, although distinct, are strongly linked. It can be a framework for designing machines and software, and it can also be a framework for understanding what these machines do. One could object to thinking of computer science with the concept of constraint, that this concept is relevant primarily for this second sense of theoretical computer science, oriented towards understanding. However, this is precisely not the issue here because a theory allowing a more precise understanding of what computers do aims at feeding practice by leaving aside a reductionist conception of computer science where only the isolated machine and its capacities would count. At the same time, the user and the programmer are considered radically unknown. Against this dichotomy, rebasing theoretical computer science thus aims at reinserting the noesis in computer science as a fundamental question for the work of engineers.

Bibliographic References

Anderson C., *The end of theory: The data deluge makes the scientific method obsolete*, «Wired magazine», 16 (7), 2008, pp. 16–07.

Bailly F., Longo G., *Mathematics and the natural sciences; The Physical Singularity of Life*, Imperial College Press, London 2011. https://doi.org/10.1142/p774.

Dwivedi D., *Through the Great Isolation: Sans–colonial*, «Philosophy World Democracy», 2020.

Keller E.F., *Refiguring life: Metaphors of twentieth–century biology*, Columbia University Press, New York 1995.

Knuth D.E., *Structured Programming with Go to Statements*, «ACM Comput. Surv.» (New York, NY, USA) 6, no. 4 (December), 1974, pp. 261–301. ISSN: 0360–0300. https://doi.org/10.1145/356635.356640.

Lecointre G., Le Guyader H., *Classification phylogénétique du vivant*, Belin, Paris 2006.

Longo G., Miquel P.–A., Sonnenschein C., Soto A.M., *Is information a proper observable for biological organization?*, «Progress in Biophysics and Molecular biology», 109 (3), 2012, pp. 108–114. ISSN: 0079–6107. https://doi.org/10.1016/j.pbiomolbio.2012.06.004.

Longo G., Mossio M., *Geocentrism vs genocentrism: theories without metaphors, metaphors without theories*, «Interdisciplinary Science Reviews», 45 (3), 2020, pp. 380–405. https://doi.org/10.1080/03080188.2020.1798588.

Longo G., Palamidessi C., Thierry P., "Randomnes: four questions and some challenges", in Zenil H. (ed. by), *Randomnes: 5 questions*, Automatic Press / VIP, Copenhagen 2010.

Lotka A.J., *The law of evolution as a maximal principle*, «Human Biology», 17 (3), 1945, pp. 167–194.

Monod J., *Le hasard et la nécessité.* Seuil, Paris 1970.

Montévil M., *Measurement in biology is methodized by theory*, «Biology & Philosophy», 34, no. 3 (April), 2019, p. 35. ISSN: 1572–8404. https://doi.org/10.1007/s10539–019–9687–x.

Montévil M., Mossio M. *Biological organisation as closure of constraints*, «Journal of Theoretical Biology», 372 (May), 2015, pp. 179–191. ISSN: 0022–5193. https://doi.org/10.1016/j.jtbi.2015.02.029.

Montévil M., Mossio M., *The Identity of Organisms in Scientific Practice: Integrating Historical and Relational Conceptions*, «Frontiers in Physiology», 11 (June), 2020, p. 611. ISSN: 1664–042X. https://doi.org/10.3389/fphys.2020.00611.

Montévil M., Mossio M., Pocheville A., Longo G., *Theoretical principles for biology: Variation*, «Progress in Biophysics and Molecular Biology», 122, no. 1 (August), 2016, pp. 36–50. ISSN: 0079–6107. https://doi.org/10.1016/j.pbiomolbio.2016.08.005.

Petit C., *L'impact de la technologie sur l'emploi, l'économie et la société (4). Entretien avec Bernard Stiegler*, «Revue du MAUSS permanente», (May), 2019.

Schrödinger E., *What Is Life?*, Cambridge University Press, Londre 1944.

Soto A.M., Longo G., Miquel P.–A., Montevil M., Mossio M., Perret N., Pocheville A., Sonnenschein C., *Toward a theory of organisms: Three founding principles in search of a useful integration*, «Progress in Biophysics and Molecular Biology» 122, no. 1 (August), 2016, pp. 77–82. ISSN: 0079–6107. https://doi.org/10.1016/j.pbiomolbio.2016.07.006.

Soto A.M., Sonnenschein C., *Information, programme, signal: dead metaphors that negate the agency of organisms*, PMID: 33100483, «Interdisciplinary Science Reviews», 45 (3), 2020, pp. 331–343. https://doi.org/10.1080/03080188.2020.1794389.

Stiegler B., *Noodiversité et technodiversité. Eléments pour une refondation économique basée sur une refondation de l'informatique théorique*, «PUnpublished», 2020.

Stiegler B., Collectif Internation, Clezio J., Supiot A., *Bifurquer: Il n'y a pas d'alternative*, Les liens qui libèrent, Paris 2020. ISBN: 9791020908575.

Turing A.M., *Computing machinery and intelligence*, «Mind», 59 (236), 1950, pp. 433–460.

Turing A.M., *The Chemical Basis of Morphogenesis*, «Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences», 237 (641), 1952, pp. 37–72. https://doi.org/10.1098/rstb.1952.0012.

Walsh D.M., *Action, program, metaphor*, «Interdisciplinary Science Reviews», 45 (3), 2020, pp. 344–359. https://doi.org/10.1080/03080188.2020.1795803.