COGNITIVE REPRESENTATION : COMPUTABLE OR NON COMPUTABLE?

Jean Guy Meunier
Université du Québec à Montréal

> *The class of problems capable of solution by the machine can be defined fairly specifically. They are [a subset of] those problems which can be solved by human clerical labour, working to fixed rules, and without understanding. (Turing 1946: 38-9.)*

published in " Representation in Science" (Aggazi E. (ed)) International Academy of Philosophy of Science , Louvain 2012.

1 The computable mind.

Be it through a variety of authors from or through various philosophical and psychological paradigms, one of the dominant concepts used in most contemporary theories of mind and cognition is the concept *representation*.

> *Representation is one of the most central concepts in cognitive science: there is no cognition without reprcesentation, and no cognitive science either" (Billman 1998: 658)*

The main classical definition of this semiotic concept stems from the classical medieval formula *Aliquid stat pro aliquo:*[1]

> *Signum est quod se ipsum sensui et praeter se aliquid animo ostendit) (Augustine 1975, 86).*

Poinsot's, one of the main influent late medieval professor defined *representation* in terms of *" being present again but under a new form. " (Poinsot, TS, II, q 1, 696b17-30)*

It is such a definition that Brentano referred to when he talked of the scholastic definition of representation. But surely the most classical and often quoted definition comes from Peirce:

> *A sign or representamen is something which stands to somebody, for something in some respect or capacity. (Peirce 1960 2. 228)*

In contemporary cognitive sciences, under different lexical clothing (*symbol, information,* or, *sign, signal i, symbol, proxies,* etc.) we find this same conceptual definition. (i. e. *the stand for* relation). And one of most classical one is given by Marr.

> *We are dealing with information -processing machines, and the way such machines work is by using symbols to stand for things to represent things"(Marr 1982: 21)*

And it was reformulated in a hundred other ways by so many authors:

> *Representation is simply another term to refer to a structure that designates. (Newell 1980 :17)*
>
> *« By representational system (RS) I shall mean any system whose function is to indicate how things stand with respect to some other object»*[2]*(Dreske 1988: 52)*

---

[1] In fact the real expression seem to have been "suppositio stat pro aliquo" see Karl Bühler: *Sprachtheorie*, Jena 1934: 40 (rpt. Stuttgart 1982); R. Jakobsen: *Essais de linguistique générale*, Paris 1963

[2]

In contemporary cognitive sciences, many researchers have recognized three important explanatory values to the concept of representation: The first one lies in the *semiotic* property of a representation implicit in the "stand for" expression of the definition. It is this property that allows cognition to be a set of intentional states that is: states that are not to be taken for themselves but for what they are "about ". The second lies in its *material independence*: that is: its nature can be explained independently of the material vehicle in which a representation can be implemented. Finally the third explanatory value lies in its *functional* role. It is by this property, that representations are understood as cognitive "states" that can be *operated* upon. Although all tree properties are important, they have raised many debates in philosophy and cognitive sciences. In this paper, we shall mainly focus on the functional role, although, as we shall see later, it is not without relation to the semantic and material dimensions. [3]

As it is well known, Fodor has been one the main defender of a representational theory of mind. He has defended the intentional status or representation but, more importantly, he has given a specific formulation to their functional role. A formulation that opened up the computational model of cognition.

> *If a mental process can be functionally defined as an operation on symbols, there is a Turing machine capable of carrying out the computation. (Fodor 1981, 130,)*
>
> In this paradigm, cognition is seen as a sort of machine where representations are "computed": Cognition operates on representations like a Turing machine. And the brain is a special type of physical instantiation of this machine.

Refusal of the representationnalist approach.

Through the years, there has been a profusion of critics[4] of this model. Many have insisted on the limits of this computational model of cognition (and mind). It is strictly a "syntactic" model of the mental operations on representation (signs, signal or symbols) and it has no explanation on the semantics of these symbols.

> *Computation is just interpretable symbol manipulation ; cognition isn't (Harnad 1994:1)*

Hence, this computational model cannot explain basic mental operations be it perception, categorization, or more deeply consciousness. As Putnam underlines it : It is a not well thought out model.

> *« Materialist are committed to the view that a human being is- at least metaphorically a machine. It is understandable that the notion of a Turing machine might be seen as just a way of making this materialist idea precise. Understandable, but hardly well thought out. (Putnam 1992: 4)*

---

[3] There are many understandings of what is a " functional role" . In this paper, we shall keep to a very general understanding: The functional dimension pertains to the description of the functional processes . And it is precisely the nature of this process that it the theoretical question explored in this paper . And depending on the understanding of what is the nature of this process follows specific understandings of the " role" , itself, if not its causal role.

[44] Among these critics: Block 1981, Putnam 1992, Maudlin 1989, Mellor 1989, Searle, (1992) Penrose 1994, van Gelder 1993 Port and Van Gelder 1997, Wright 1995, Horst 1996, Copeland 2000, Fetzer 2001

It cannot explain neither the emergence nor the acquisition of representations themselves. For some[5]this meant that the concept of representation itself is inadequate for modeling cognition.

It is the concept of representation, which is insufficiently sophisticated. (Van Gelder 1993 : 6).

Representation is the wrong unit of abstraction in building the bulkiest parts of intelligent systems. (Brooks R. A. 1991: 396).

We are not building representations at all! (Thelen et Smith 1994 : 338).

However, some others researchers have hinted elsewhere. Maybe the concept of representation is not the problem. Maybe it is the concept of *computation* itself that is inadequate. And in this line of thought, a first type of such critics claims that this concept of *computation* cannot deal with the complexity of mind and cognition.

That the kind of activity involved in the execution of mundane procedures seems to involve thinking yet reflects a class of effective procedures which does not qualify as Turing-computable. " (Fetzer 2001:116)

Human intuition and insight... cannot be reduced to any set of computational rules. (Penrose, Shadows of the Mind: 65

A second type of critics more radical claims that it is the concept of computation itself that is problematic and our understanding of it should be requestionned:

... the classical Turing paradigm may no longer be fully appropriate to capture all features of present-day computing. - J. van Leeuwen and J. Wiedermann 2000: 22

It this paper, our interest is to explore this second type of critics which invites a revisiting of the concept of *computation* itself. We aim to see how this revisiting has an impact on the representational aspect of cognition.

2. The computable

Although the concept of computation seams intuitively clear, its real meaning is not as transparent as we may think. Although the Latin word "computare" has been traditionally used by many philosophers for talking about the rationalization process of mind, the modern meaning of *computation* takes its source in the enquiry into the question of *calculus* as explored from Descartes, Leibniz, Peano, Dedekind up to Hilbert, Godel, Church. Etc. But it is only with Turing, if not with Kleene that the term "*computation*" itself has been used and accepted in its contemporary sense: a process of effective procedure for solving arithmetic problems. And its main understanding has been given through synthesis of two mathematical thesis: the Church thesis and the Turing thesis.

The Church thesis : 1938 « Every effectively calculable function (effectively decidable predicate) are is general recursive. "

The Turing Thesis is:… " *every intuitively computable function is computable by an abstract automata"*

These two related thesis form a unified thesis called the Church-Turing thesis (CTT). And the usual understanding of this later thesis was extended in such a way as to make the concept of *computation* practically synonymous with a set of other interrelated concepts

[5] See Winograd et Flores 1986 ; Moravec,1988 ; Varela, 1988 ; Chalmers 1996; Port and Van Gelder, 1995; Petitot 1985; Scott-Kelso 1995; Brooks, 1991 ; Franklin, 1995 ; etc.)

such as *decidable sentences*, *recursive function*, *effective procedure*s, *algorithms*, *programs*, *Post production*, *markovian systems, etc.*

But we must be careful on the formulation of this CTT. Strictly speaking this thesis does not declare that the preceding terms are synonymous. It says only that these terms have and identical reference, that is, they are equivalent from an extensional point of view.

> *« The same class of partial functions (and of total functions can be obtained in each case (Rogers 1987:39)*

Still, the union the Church thesis and the Turing thesis have given the traditional accepted definition of computation.

Recently some historians of mathematics and finer grained mathematicians have come to raise doubts on the synonymicity or intensional identity of these various terms contained in the Church-Turing thesis. In fact, there are raising questions about the real content of this famous thesis iself. For the Church-Turing thesis does not ware its meaning on its sleeve, as Israel, paraphrasing Ayers, cogently declares :

> *This Church-Turing thesis hardly wears its meaning on its sleeve and differing conceptions (and misconceptions) of computation may lie behind what seems a widespread consensus. (Israel 2002 : 181)*

In other words, the meaning computation is not as evident as one may think. And some believe that we should requestion the adequacy of this thesis for understanding present day computing.

> *The classical Turing paradigm may no longer be fully appropriate to capture all features of present-day computing. ”- (J. van Leeuwen, J. Wiedermann 2000: 22)*

> *It makes no sense to point to Turing computation as the `only true model of computation'. Rather, there are many models of computation, each making sense in its own domain.. Stannett 2006: 16*

More so, some see the classical theory of computation as a failed program :It is a

« last vestige of a failed program »(Cleland 2007)

Although highly radical, these critics require attention in the context of a theory of cognition and mind. They attack directly the foundations of the computational representational model of cognition.

3-The computable and the non-computable.

Even if the CTT maintains an extensional identity between the various terms defining the concept of computation, there are some important differences between these terms that invite us into a deeper understanding of what is “computation”. Indeed, the semantic field of this concept contains many conceptual differences that, often are at the heart of mathematical debates over what is computation. But, as we shall see, the differences are not easy to unveil. One place where we can look to find these minute differences is in the definition given by the original creators of the theory of the opposite concepts of computation, that is : in the *non decidable*, the *non algorithm and the non computational*, etc...

For instance, Hilbert, as Davis says, had though that he could settle all mathematical problems thought decidable procedures:

" it seemed clear to Hilbert that with the solution of this problem, the Entscheidungsproblem, it should be possible, at least in principle, to settle all mathematical questions in a purely mechanical manner. (Davis 1965, p. 108)"

But he did not manage to prove this for all arithmetic. He was then confronted to radical *undecidability* of a problem that he reformulated in terms of Entscheidung problem that is : is arithmetic really decidable?

The same goes for Godel and Church. One of the main contribution of these mathematicians on the question of computation was in a sense a negative one. Although both deepened the concept of *calculus* with the concepts of *recursivity, equational calculus* and *lambda calculus*, Godel's main contribution was to give a proof that arithmetic was not-decidable that is : it is incomplete[6]. Later, Church proved again that this Entsheidung problem could not be recursively definable or more technically or receive a lambda defintion. These positions were formulated as *thesis*, not as *theorems*.

More so, as Davis (1982) [7]and Sieg (1994), underlined it, Gödel was not convinced that *recursivity* was an adequate definition of " effective procedure" or decidability. Something was missing: For Godel more specification was needed on what exactly should be understood by *recursion* as means for *calculation*.

Turing's own contribution on this topic is most interesting. In the continuity of Gödel and Church, he presented in 1934 a model that translated what were only "intuitive" steps in the recursion model into specific mechanical procedures. These mechanical procedures were realized in what he called an "abstract automata". This automata, as we all know could itself be realized in a physical machine incorporating a moving tapes on which symbols taken from a finite set of symbols can be written or erased according to a finite set of instructions. And Turing called the function (see Sieg, 1994 :95) that these automata could process " a computable function" " *human calculating with a pencil and paper"* and he named these humans *"computor "*. [8] But it is only later that Kleene called theses operations :"effective elementary operations of " computation". [9]

It may be a surprise for many that the real interest of Turing was not to give a definition of what makes a function *computable* but what make it *not computable*. One must remember that at that time Turing, as much Church was entangled in the Gödel's incompleteness theorem. And as Hodge says, his answer to this question would allow him,

---

[6] Let κ be any recursive consistent class of FORMULAS; then the SENTENTIAL FORMULA stating that κ is consistent is not κ-PROVABLE ... (Godel 1931 p. 614)

[7]. *Gödel did* not *think of general recursiveness as a rigorous explication of effective calculability… The conjecture stated there only refers to the equivalence of 'finite (computation) procedure' and 'recursive procedure'. However, I was, at the time of these lectures, not at all convinced that my concept of recursion comprises all possible recursions; …(*Davis 1982*:8)*

[8] *A man provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine (Turing 1948:9).*

[9] *Turing's machine concept arises from a direct effort to analyze computation procedures as we know them intuitively into elementary operations. Turing argued that repetitions of his elementary operations would suffice for any possible computation.\ For this reason, Turing computability suggests the thesis more immediately than the other equivalent notions and so we choose it for our exposition."(Kleene, 1967: 233)*

- he hoped- just as Godel and Church had done before him, to prove the unsolvability of to the Entsheidungsproblem.

> *…a definitive negative answer to Hilbert's* Entscheidungsproblem. *Turing showed it possible to give unambiguous definitions of real numbers which are* not computable (Hodges 2000)

And most importantly, it was this problem of the unsolvabilty of mathematical problems through these mecanical procedures that he translated into to his own view of it: the *halting problem* which one the most important *non-computable* problem.

In summary, the specific positive contribution of Turing gave more precisions to the concept of *calculabiliy* and *recursion*. It translated concepts that were still too intuitive into a set of atomic, local and structured procedures. And because of these procedures could become mechanical, they hence were *effective*. Godel saw this as the specific contribution of Turing.

> *With this concept one has for the first time succeeded in giving an absolute definition of an interesting epistemological notion, i. e., one not depending on the formalism chosen. In all other cases, such as demonstrability, one has been able to define them only relative to a given language... For the concept of computability, however, although it is merely a special kind of demonstrability or decidability, the situation is different. (Gödel 1990a: 150)*

But this contribution, opened up a new problem: if computation is to be defined as set of " well defined effective procedures" even more " of "mechanical procedures", what does it mean to process *non-computable* functions: a question which Godel himself considered a very complex and fundemental problem for mathematics.

> *It must be admitted that the construction of a well-defined procedure which could actually be carried out (and would yield a nonrecursive number-theoretic function) would require a substantial advance in our understanding of the basic concepts of mathematics. (Gödel 1990b; 308).*

The Turing and Post solutions.

How do you deal with non-computable functions? Answering this queston is a Pandora's box and it is at the heart of the enrichment of contemporary theory of computation. Here we can only have a peek in the Turing and Post solutions to this question and relate it to our problem of cognition and representation. Others (Chatin, Da Costa, Doria , 2012) explore other original paths but the relation to cognition is not yet that evident .

Turing's technical answer to the processing of non-computable functions was to transform his classical machine " abstract automata" (A-Machine) into a machine (O-machine) that could be assisted by some Oracle: [10] But as many commentators have

---

[10] *A Turing oracle machine (o-machine) is a Turing machine with an extra \read only" tape, called the oracle tape, upon which is written the characteristic function of some set A (called the oracle), and whose symbols cannot be printed over. The old tape is called the work tape and operates just as before. The reading head moves along both tapes simultaneously. … The Turing oracle program* P_e

underlined it, Turing introduced this concept or Oracle with practically no details in a tiny and obscure place of his 1939 paper.

Technically translated: an oracle is a machine that is in itself a computable function (often it is a characteristic function). And, when it is necessary, it will intervene in the processing of another machine (called the A-machine) so that a solution can be found.

> *An oracle machine (o-machine): roughly a Turing a-machine which could interrogate an \oracle" (external database) during the computation. (Soares 2009: 23)*

In this manner, this O-machine can process any non-computable function. It allows any computation to halt. Still later on, in 1944, Turing slowly moved towards a more formal definition of the oracle: He saw it as sort of meta-computable function that could check lower order functions. As Turing says in a letter to Newman:

> *One imagines different machines allowing different sets of proofs, and by choosing a suitable machine one can approximate 'truth' by 'provability' better than with a less suitable machine, and can in a sense approximate it as well as you please (Gandy 1954 : 22)*

Slightly later, he proposed that certain non-computable functions of a lower level could be solved in a logic of one degree higher. And it seems that he thought that this would counter the Godel theorem. A proposition that it seems he later abandoned. More so, that Feferman (1962) proved incorrect.

A general incompleteness theorem for recursive progressions... would have been dramatic proof of the far-reaching extent of incompleteness phenomena. However, the situation has not turned out in this way. Feferman 1962: 258

Still, even if these propositions were proven later to be incorrect, they opened up the idea that non-computability could be relative. That is : computation and non-computation are not absolute notions : More profoundly there seems to be "degrees of solvability" in proof constructions or "effective reducibility" [11]as Post later in 1944

> *For unsolvable problems the concept of reducibility leads to the concept of degree of unsolvability, two unsolvable problems being of the same degree of unsolvability if each is reducible to the other, one of lower degree of unsolvability than another if it is reducible to the other, but that other is not reducible to it, of incomparable degrees of unsolvability if neither is reducible to the other. (E. L. Post 1944: 289)*
>
> *"... This transformed the notion of computability from an absolute notion into a relative one that would lead to entirely new developments and eventually to vastly generalized forms of recursion theory". (Feferman 2006:1204)*

Unfortunately, as Soares (2009) underlines it, these concepts of *oracle* and *relative computability*, when they were published, were not really well understood and received[12].

---

takes some oracle A and defines a partial A-computable functional $\varphi_e^A(x) = y \ \}$*" Soares :2009: 21*

[11] *In 1944 Emil Post [20] took it as his basic notion for a theory of degrees of unsolvability, crediting Turing with the result that for any set of natural numbers there is another of higher degree of unsolvability. This transformed the notion of computability from an absolute notion into a relative one that would lead to entirely new developments and eventually to vastly generalized forms of recursion theory.* Feferman 2006: 1204

[12] There is a relation here to be explored with Kripke's possible worlds.

At that time, it was mainly the Church-Turing thesis that retained attention because of the emerging spectaculer computer technology.

> *It is surprising that so much attention has been paid to the Church-Turing Thesis. Over the last seventy years and so little to the Post-Turing Thesis 6. 1 on relative reducibility, (Soares 2009: 23)*

It was only later that these concepts were seriously reintroduced. But only a few mathematicians saw the importance of the concept of relative unsolvability for an adequate theory of computation.

Still Rogers in 1967 wrote on the subject still using the Turing original definition of 1939. Even now, it seems not very well explored (see Odifreddi 1989, Lerman 1983). Certain researchers such as Cooper 2004, and mainly Soare (2009) insist more and more on the importance of this question of non-computation.

> *The subject (of computation) is primarily about incomputable objects not computable ones, and has been since the 1930's. The single most important concept is that of relative computability to relate incomputable objects. (Soare 2009: 59)*
>
> *Everyday mathematics leads us unavoidably to incomputable objects. (Cooper 2004: 1)*
>
> *" undecidability and incompleteness are everywhere in mathematics" Chatin, Costa , Doria; 2012: xiii)*

Many sub problems, often highly technical were and are still explored in various sub fields of what still is named "theory of computation" even if many of the problems dealt with would classically fall into the " uncomputable" problems. [13]Typical exemples are the *word problem*, the "productive sets of the diophantise equations".

But the question of non-computation was recently awaken by Penrose (1989) and Kieu (2008) who explored it in the quantic field. And other types of physical foundations have been explored under the name of *hypercomputation*[14]: A subfield that has taken many rich but highly debatable orientations. But we will not pursue them here.

> *« Hypercomputation typically refers to systems that can compute non-recursive functions, but one also talks of super-Turing systems, which do not*

---

[13] *We should use the term \recursive" to mean \defined inductively" not \calculable" or \computable." The subject is called \Computability Theory" not \Recursive Function Theory" or \Recursion Theory."(Soares, 2009: 58)*

[14] Here are a few more of these definitions of hypercomputation.

Hypercomputation is the computation of functions or numbers that cannot be computed in the sense of Turing i.e. cannot be computed with paper and pencil in a finite number of steps by a human clerk working effectively. (Copeland :2002: 489

Hypercomputation theory is the theory of any device or devices whose calculating properties exceed those of any Turing machine, or equivalently which exceeds the computing power of partial recursive functions, formalconstructs which derive from the above-mentioned papers by Kleene, Church and Turing. Doria and all 2006: 1

Hypercomputation typically refers to systems that can compute non-recursive functions, but one also talks of super-Turing systems, which do not necessarily compute anything non-recursive, but which nonetheless outperform Turing machines in terms of complexity or other measures Stannett 2006:9

Computation fails as an explanatory notion for mind, the critics claim, because computation, assumed to be defined solely in abstract syntactic terms, necessarily neglects the real-time, embodied, real-world constraints with which cognitive systems intrinsically cope. (Scheutz , 2002: 4)

necessarily compute anything non-recursive, but which nonetheless outperform Turing machines in terms of complexity or other measures (Stannett 2006, 8)

Soare's interpretation

Recently, Soare (1996, 2006, 2009 etc.) -an important contributor on the theory of computation- proposed a relatively original understanding of the notion of oracle and relative computability for a theory of computation. One that can explain how a computer can easily compute " a non-computable function "! And this understanding gives quite a renewal of the role of the computational model for theories of cognition.

According to Soare, the Oracle in a Turing machine is itself a partial computable function. [15] Its input comes from a A-Machine which is processing a computable or non-computable function. And this oracle function itself, instead of being defined recursively, can be defined extensionally that is by the means of its graph. So, when a A-machine is processing a non-computable function and it encounters a problem for one of its input value-x, it can consult the Oracle by presenting it this value x. And the oracle can give its own answer-y to this value-x. An answer that it found in the graph of the oracle function which is a list of the possible solutions given in a form of a list where each x_i is paired to some value y_i. And this answer of the Oracle allows the A-machine to continue and eventually to halt.

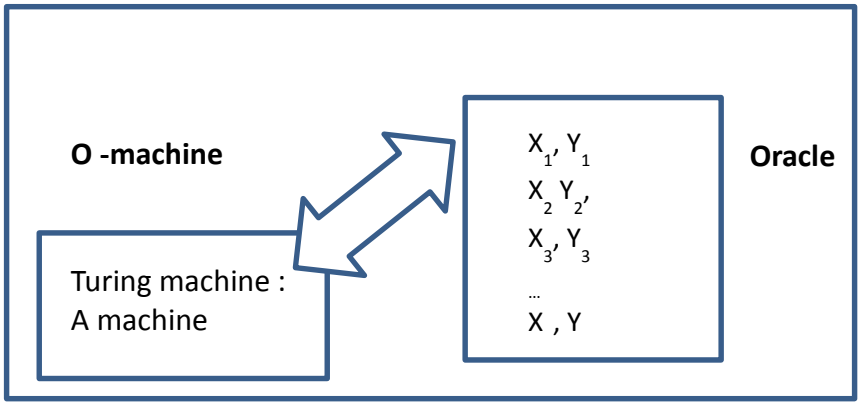

FIGURE I

Figure 1 : The Oracle as a list in a O machine

Now, from another point of view we can see this list as an external data base that any A-machine can consult. More so, this external database can itself be seen as a possible environment in which they A machine operates and can consult.

---

[15] The Turing oracle program P_e takes some oracle A and defines a partial A-computable functional $\varphi_e^A(x) = y \quad \}$" Soares :2009: 21

> *In real world computing the oracle may be called a database or an environment. "the oracle may be called a database or an environment. " (Soares 2009: 41)*

One immediate consequence of this reading of the Oracle as a data base, is that the combination of the O-machine with an A-machine is richer than the classical A-machine. A computational machine is no more a closed machine. An A-machine can now can consult or interact with something different than itself in order to continue its processing.

This is where this interpretation becomes interesting for a theory of cognition. The data base-oracle, being external to the A-machine can be anything where is stored information that can be consulted by an A machine! And one of these external oracles can simply be embedded in the environment or any other form of informational sources. It can for instance be in an external hard disk memory, be on line, or even more simply be in another machine in its environment.

In summary, in this model of computation, a Turing machine, that is, an A-machine, is now changed into a O-machine which is an A-machine plus a Oracle. The A-machine can always process a classical computational function. But if it is confronted to a non-computable, or non-recursive function, it may the consult the oracle so that a solution can be found. And this oracle can be external to the A-machine.

Relation to representation

Let us now come back to our original problem: the question of representation. Can this refurbished concept of computation have any impact on the *computational theory of cognition?* We can briefly see two impacts of this refurbished model.

The first impact pertains to the "openness" of cognition. In this perspective, if a computational A+0 machine is not a closed system as is an A-machine, then, this allows to model cognition as an open system. A cognitive agent can, in a problematic situation call upon an "Oracle" which in turn can take many forms: It could be a an internal memory built out of past experiences. But it could also be *evolution*. More so it could the body, or any other artifacts. The means that the brains may not be the only material carrier of information on which cognition may rely. More so, a mind could interact with environment in which it finds solutions to some of the problems it seeks to solve. And this environment is not limited to "physical" nature. It can also take the form of be others minds! In other words, a cognitive agent, not being a closed system may inhabit a word with which it interacts. And this world can be nature but it could also be other human beings.

The second impact pertains to the "flexibility " of rationality that a theory of cognition should offer. In this perspective, an A+O-machine does not require the processing of information to be as stringent as the classical model is : that is it does not require pure atomicity, systematicity, productivity, sequentiality. If there is some cognitive operation that can be seen as " language" (Fodor 1975) that is a " regular language" or a Turing machine, the O machine allows also to see that many operations can deal with "exceptions". More so, it could process by trial and error, as Putnam (1965) saw it. For Gandy (1980), computation may even be parallel, allowing then new emergent dynamical connections and forms. In other words, a computational model of cognition is not limited to process only discrete and linear inputs. Hence, the systematicity and productivity are not essential dimensions of computation as Fodor and Pylyshyn have maintained. More so, if an O-machine is not constrained to only pure classical "computational" tasks that is, if it

can also deal with "non-computational" ones, this means that cognition would not be limited to pure "grammatical" operations. In other words, its operations are not necessarily always grounded only in a pure "rationality". [16]Chances are that these types of operations are common. We believe that they are the most important and commons tasks a mind has to deal with in a daily basis.

Still, there are some questions that the refurbished theory of computation seems to have difficulty coping with : How are these representations or symbols in a list themselves grounded (Harnad 1990)? How can it help us understand the question of consciousness, self-identity etc. Bringjord and Zenzen (2005) seem to believe that it offer pertinent answers.

In conclusion, because it gives place to relative computability, this revisited computational model of cognition may help us understand mind as something else than a closed automatic system embedded in algorithms and grammars. It also gives place to creativity, natural interaction, trial and error learning. Because it can deal with both with "computable" and "non-computational" tasks it can also help us better a better understanding of rationality but also of irrationality.

.

Augustine. *(1975 ed). De Dialectica. trans. D. Jackson. Synthese Historical Library, Reidel.*

Cooper S. B, (2004). Computablility Theory, Chapman & Hall/CRC Mathematics, London, New York.

Billman D. (1998). Representations. In: *A Campanion to Cognitive Science,* W. Bechtell and G. Georges (eds), Blackwell, pp. 649-659.

Block N. (1981). Psychologism and behaviorism. Philosophical Review 90:5-43.

Block Ned. (2000). *"Introduction: What is Functionalism?". In Readings in Philosophy of Psychology vl. ed. Ned Block. Klein 2000: 171-184.*

Boolos G. and R. Jeffrey, (1974). *Computability and Logic,* Cambridge Univ. Press, Cambridge, Engl.

Bringsjord S. (1998). 'Philosophy and 'Super' Computation', in J. Moor and T. Bynam, eds.,(1998). *The Digital Phoenix: How Computers are Changing Philosopy*, Oxford: Blackwell, pp. 231–252.

Bringsjord, S. M. Zensen,(2005). *Superminds,* Springer, Brooks, R. A. (original 1991,) *Intelligence without representation. In Haugeland, J., ed., Mind Design II. Cambridge, MA: MIT* Press, pp. 395-420

Bühler K. (1934): *Sprachtheorie*, Jena (rpt. Stuttgart 1982);

Chalmers D. J. (1996). *Minds, machines, and mathematics*. Psyche 2:11-20.

Chatin, G., Da Costa N., Doria,A.F. (2012) *Gödels'Way* . CRC Press

Cleland Carol E. (2007) The Church—Turing Thesis. A Last Vestige of a Failed Mathematical Program In Adam Olszewski, Jan Wolehski, Robert Janusz. (Eds.) *Church's Thesis After 70 Years.*

[16] The Doria, Costa, Chatin (2012) reseach on the the undecidability of chaoting sytems may be a source of renewal for understanding cognition.

Cooper, S. B. (2004). The incomputable Alan Turing. *British Computer Society, Electronic Workshops in Computing.* Invited paper from 'Alan Mathison Turing 2004: A celebration of his life and achievements, Manchester University, 5 June.

Copeland, B. J. (2000). 'Narrow Versus Wide Mechanism', *Journal of Philosophy* 96, pp. 5–32.

Copeland B. J. (2002). Hypercomputation *Minds and Machines* 12: 461–502, 2002.

Davis M. (ed.) (1965). *The Undecidable*, Hewlett, NY: Raven Press.

Davis M. (1982).*Computability and Unsolvability* ,Dover.

Doria Francisco Antonio, José Félix Costa (2006). *Introduction to the special issue on hypercomputation* Applied Mathematics and Computation 178 (2006) 1–3 Mathematics and Computation 178 (2006) 1–3.

Dreske, F. (1988). *Explaining Behavior. Reasons in a World of Causes*, MIT Press, Bradford Book.

Feferman, S. The impact of the incompleteness theorems on mathematics, *Notices Amer. Math. Soc.* 53 (April 2006), 434–9.

Feferman, S.(1962) Transfinite recursive progressions of axiomatic theories, *J. Symbolic Logic* 27 (1962), 259–316.

Fodor, J. A. (1981). *Representations*. Cambridge: MIT Press.

Fodor J. A. and Pylyshyn, Z. W. (1988)Connectionism and Cognitive Architecture: A Critical Analysis. Ó Cognition 28 (1-2), 3-71. Fodor, J. A. (1975) The language of thought New York: Thomas Y. Crowell.

Franklin S., (1995). Artificial Minds, MIT Press.

Gandy R. (1980). 'Church's Thesis and Principles for Mechanisms', in K. Barwise and Kunen, eds., *The Kleene Symposium,* North-Holland, pp. 123–148.

Gandy R. O. (1954). Letter to M. H. A. Newman, in the Turing Archive, King's College, Cambridge, included in the *Mathematical Logic* volume of *The Collected Works of A. M. Turing,* eds. R. O. Gandy and C. E. M. Yates (Amsterdam: North-Holland).

Godel, K. (1931),.'On Formally Undecidable Propositions of Principia Mathematica and Related Systems', in Van Heijenoort (1967), pp. 596–616.

Godel, K, (1990a).'Remarks Before the Princeton Bicentenial Conference on Problems in Mathem-atics', in S. Feferman, J. J. W. Dawson, S. C. Kleene, C. H. Moore, R. M. Solovay, and J. van Heijenoort, eds., *Kurt Godel: Collected Works,*

Godel K. (1990b). 'Some Remarks on the Undecidability Results', in S. Feferman, J. J. W. Dawson, S. C. Kleene, G. H. Moore, R. lvi. Solovay, and J. van Heijenoort, eds., *Kurt Godel: Collected Works, Vol. II,* New York: Oxford University Press. With *introductory material by Judson Webb*.

Godel K. (1995 *) Collected Works,* ed. S. Feferman et al., vol. iii (Oxford: Oxford University Press.

Harnad S (1994). Computation Is Just Interpretable Symbol Manipulation: Cognition Isn't http://cogprints. org/1592/

Hodges A, 2000 *Uncomputbiity in the work of Alan Turing and Roger Penrose* Lecture http://www. turing. org. uk/philosophy/lecture1. html

Horst S. W. (1996). *Symbols, computation and intentionality : a critique of the computational theory of Mind,* U. of Cal. Berkeley Press

Israel D. (2002). 'Reflections on Gödel's and Gandy's Reflections on Turing's Thesis', *Minds and Machines* 12, pp. 181–201.

Jakobson R. (1963). *Essais de linguistique générale*, Paris.
Fetzer J.H. *(2002). Computers and Cognition: Why Minds Are Not Machines, Dor-drecht: Kluwer, 2001, xix + 323 pp*.
Kieu Tien D (2008) *Computing the noncomputable*, Centre for Atom Optics and Ultrafast Spectroscopy, Swinburne University of Technology, Hawthorn 3122,
Kleene, S. C 1967. *Mathematical Logic.* John Wiley. Dover.
Lerman, (1983) M. *Degrees of Unsolvability*: *Local and Global Theory*, Springer-Verlag, Heidelberg New York Tokyo, .
Marr, D (1982), Vision, San Francisco, Freeman.
Maudlin, T. (1989). Computation and Consciousness. *The Journal of Philosophy*, pages407-432
Mellor D. H (1989). How much of the mind is a computer. In (P. Slezak, ed) Computers, Brains and Minds. Kluwer.
Moravec H. (1988). *Mind Children: The Future of Robot and Human Intelligence*, Cambridge MA:Harvard University Press.
Newell A. (1980). Physical Symbol Systems. *Cognitive Science*, 4, 135-183. 17
Odifreddi P., *Classical Recursion Theory*, North- Holland, Amsterdam, Volume I 1989, Volume II 1999.
Peirce, C. S. *: Collected Paper from Charles Sanders Peirce. vol 2 (Cambridge Mass: Harvard University Press. 1960 2. 228*
Penrose, R. (1989), *The Emperor's New Mind*, Oxford University Press
Penrose, R. (1994) Shadows of the Mind: A search for the Missing Sicence of Consciousness, Oxford University Press.
Petitot, J. (1985b) *Morphogenése du Sens*. Paris: Presses Universitaires de France
Poinsot J. (1985) *Tractatus de Signis*. in *ex Cursus Philosophicus. <1631-1635>* John Deely (ed). Berkekey: University of California Press
Port R. F., and van Gelder (1995) T., Eds., *Mind as Motion*. Cambridge, MA: MIT Press, pp.
Post E. L., Absolutely unsolvable problems and relatively undecidable propositions: Account of an anticipation. (Submitted for publication in 1941.) Printed in Davis [1965, 340{433].
Putnam H. (1960). 'Minds and Machines', in S. Hook, ed., *Dimensions of Mind*, New York: New York University Press.
Putnam H. (1965). 'Trial and Error Predicates and a Solution To a Problem of Mostowski', Journal of Symbolic Logic 30(1), pp. 49–57.
Putnam H. (1992). *Renewing Philosophy*, Cambridge, MA: Harvard University Press.
Rogers, H. (1967), (ed 1987) *Theory of Recursive Functions and Effective Computability*, Series in Higher Mathematics, New York: McGraw-Hill
Scheutz M. (ed) (2002) *Computationalism*, New Directions.
Scott Kelso J. A. (1995). *Dynamic Patterns*, MIT Press.
Searle J. (1992). *The Rediscovery of the Mind*, Cambridge, Mass. MIT Press.
Sieg, W. (1994), 'Mechanical Procedures and Mathematical Experience', in A. George, ed., *Mathematics and Mind*, Oxford: Oxford University Press.
Soare, R. I. (2009) _Turing Oracle Machines, Online Computing, and Three Displacements in Computability, Theory, http://www. people. cs. uchicago. edu/~soare/History/turing. pdf (2011)

Soare, R. I. Computability and Recursion Author(s): The Bulletin of Symbolic Logic, Vol. 2, No. 3 (Sep., 1996), pp. 284-321 Published by: Association for Symbolic Logic

Stannett, M. (2006), The case for hypercomputation, Applied Mathematics and Computation 178 (2006) 8–24

Thelen, E & Smith, L. B (Eds) (1994) *A dynamic systems approach to development: Applications* (Cambridge, MIT Press)

Turing 1940 Letter from Turing to Max Newman, written at The Crown, Shenley Brook End, circa 1940

Turing, A. (1939), 'Systems of Logic Based on Ordinals', Proceedings of the London Mathematical Society (Series 2) 45, pp. 161–228.

Turing, A. M. 1948. `Intelligent Machinery'. National Physical Laboratory Report. In Meltzer, B., Michie, D. (eds)

Van Gelder (1993). What might cognition be, if not computation, *The journal of Philosophy*. 92, 245-381

Van Leeuwen, J. and Wiedermann, J. (2000a) *Breaking the Turing Barrier: The case of the Internet*, Techn. Report, Inst. of Computer Science, Academy of Sciences of the Czech. Rep., Prague

Varela F. J. (1988), Invitation aux sciences cognitives. Paris, Seuil

Winograd, T. and Flores, F. (1986)Understanding Computer and Cognition, Norwood. N. J. : Ab lex

Wright, C (1995) 'Intuitionists Are Not (Turing) Machines', Philosophia Mathematica 3, pp. 86–102.