Non deterministic computation over the real numbers

Gilles Dowek*

We have defended in [2, 3] the idea that the physical Church thesis, if it holds, explains the efficiency of mathematics to describe the laws of nature. And also that it suggests that the laws of nature can be expressed, not only in a language of propositions, but also in a language of algorithms.

The arguments of Gandy [4], that support the physical Church thesis, even suggest that these algorithms may operate on the elements of a countable set, such as the natural numbers, the rational numbers, the elements of an extension of finite degree of the field of rational numbers, the computable real numbers, ... Nevertheless, the desire to remain as close as possible to the traditional formulations of the laws of nature rather leads to try to describe these laws using algorithms operating on real numbers. This use of algorithms operating on real numbers may be just a first step, but it may also simplify the formulation of these laws, that might be much more complex, if formulated as algorithms operating on the elements of a countable set.

An algorithmic formulation of the laws of nature should not predict more than a propositional formulation of the same laws. As some of the laws of physics are non deterministic, it seems that we must use non deterministic algorithms to express these laws in an algorithmic way.

The notion of non deterministic algorithm and that of algorithm operating on real numbers have been defined and well studied. It seems however that the conjunction of these two notions, that is the notion of non deterministic algorithm operating on real numbers, remains to be defined. We discuss in this paper what such a definition may be.

We assume that the notion of computable function operating on the natural numbers is known. The set of such functions can, for instance, be defined as the smallest set containing the projections, the null functions, the successor function and closed by composition, definitions by induction and minimization. Other definitions based on the notions of rewrite systems, terms of lambda-calculus, or Turing machines are possible. This notion extends to countable sets by numbering their elements.

*École polytechnique and INRIA, LIX, École polytechnique, 91128 Palaiseau Cedex, France. gilles.dowek@polytechnique.edu, http://www.lix.polytechnique.fr/~dowek.

1 Non deterministic algorithms

1.1 History

The notion of non deterministic algorithm is as old as computability theory itself, as one of the first languages used to express computable functions, the lambda-calculus, has a non deterministic evaluation process: the term $(x \mapsto a)\ ((y \mapsto y)\ b)$, for instance, reduces to $(x \mapsto a)\ b$ and to a. The confluence theorem of Church and Rosser shows however that, although it is locally non deterministic, this process is globally deterministic: the reduced form of a term, if it exists, is unique.

More generally, a rewrite system defines an algorithm that is locally non deterministic as, in a term that contains several reducible subterms, it is possible to reduce any of these subterms and sometimes the same reducible subterm may be reduced by several rules. The general study of confluence of rewrite systems by Newman and then Knuth and Bendix has shown that some systems were confluent, and thus globally deterministic, and others were not. For instance, the algorithm defined by the rewrite rules $x + x \longrightarrow 2 \times x$ and $x + 0 \longrightarrow 0$ is non deterministic, as the term $0 + 0$ reduces to two distinct irreducible terms: 2×0 and 0. But the algorithm obtained by adding a third rule $2 \times 0 \longrightarrow 0$ is globally deterministic.

In the case of lambda-calculus the non-determinism is not really intended and it may be eliminated, for instance by defining a strategy. It plays, in contrast, a more central role, in the definition of asynchronous parallel algorithms, like communication protocols, or in that of randomized algorithms.

In complexity theory, finally, the notion of non deterministic Turing machine has permitted to define an important complexity class, between the class of polynomial algorithms and that of exponential algorithms: the class NP of non deterministic polynomial algorithms.

1.2 Functions and relations

A deterministic algorithm, that takes an element of a set A as an argument and returns an element of a set B if it terminates, defines a partial function from A to B. The goal of the computability theory is to characterize and study these functions, called *computable functions*, by abstracting from the execution modalities of these algorithms.

A non deterministic algorithm, in contrast, defines a relation between the sets A and B and it is therefore natural to try to characterize the relations that correspond to these algorithms. Two approaches can be used to characterize these functions. The first bases the notion of computable relation on the notion of computable function. The second tries to define, in a first step, the execution modalities of a non deterministic algorithm, to characterize, in a second step, the relations defined by such algorithms. These two approaches yield the same result: the characterization of these relations as *semi-decidable* relations or, equivalently, *effectively enumerable* relations.

1.3 Extending the notion of computable function

A relation R between two sets A and B can always be seen as a function from A to the powerset of B: as the function that maps the element x to the set R_x of the elements y of B such that $x\ R\ y$. The leads to raise, in a first step, the question of the representation of sets with computable functions.

Finite sets

If B is a countable set, the set of finite subsets of B is also countable and we can represent a finite set C of elements of B with a natural number. Thus, we can define a notion of *finitely computable* relation by stating that a relation R is finitely computable if for each x, the set R_x is finite and the function that maps each element x to a natural number representing the set R_x is computable.

Characteristic functions

When, as it is often the case in physics, the set R_x is not restricted to be finite, we can try to represent it by its characteristic function. A set whose characteristic function is computable is called *decidable*. For instance, the set of even numbers is decidable because the function that maps a natural number to the value 1 or 0 depending on the parity of this number is computable.

Using this representation of sets leads to represent the relation R by a function f that maps x to a function that maps y to the value 1 of 0 depending on the fact that $x\ R\ y$ or not. Using the fact that a function taking values in a functional space is equivalent to a function of several arguments, the relation R can also be represented as the binary function that maps (x, y) to 1 or 0 depending on the fact that $x\ R\ y$ or not, that is by its own characteristic function.

We can therefore define a notion of *decidable relation*, by stating that a relation is decidable if its characteristic function is computable.

Partial characteristic functions

An other possibility is to represent the set R_x with its partial characteristic function, that is by the function that takes the value 1 at y if y is a member of R_x, but is not defined at y otherwise. A set whose partial characteristic function is computable is called *semi-decidable*.

Using this representation of sets leads to represent the relation R by a function f that maps x to the partial function that maps y to the value 1 if $x\ R\ y$ and that is not defined at y otherwise. Using the fact that a function taking values in a functional space is equivalent to a function of several arguments, the relation R can also be represented that maps (x, y) to 1 if $x\ R\ y$ and that is not defined at (x, y) otherwise, that is by its own partial characteristic function.

We can therefore define a notion of *semi-decidable* relation by stating that a relation is semi-decidable if its partial characteristic function is computable.

Enumeration functions

The last possibility is to represent the set R_x with a computable total function whose image is this set. A set that is the image of a computable total function is called *effectively (or recursively) enumerable*. The elements of the domain Ω of this function are called *indices*. When this set is finite or countable, by using a numbering of its the elements, we can restrict, without loss of generality, to the case where it is the set of natural numbers. To avoid to distinguish a special case for the empty set, we shall consider that this function may fail, that is take a special value that is not taken into account in its image.

Using this representation of sets leads to represent the relation R by a function f that maps x to a function f_x such that $x\ R\ y$ if and only if there exists an index i such that $y = f_x(i)$. Without loss of generality, we can assume that all the functions f_x share the same set of indices. Using the fact that a function taking values in a functional space is equivalent to a function of several arguments, the relation R can also be represented by a binary function f such that $x\ R\ y$ if and only if there exists an index i such that $y = f(x, i)$.

We can then define a notion of *effectively enumerable* relation by stating that a relation R is effectively enumerable if there exists a set Ω and a computable function f from $A \times \Omega$ to B such that for all x, the set R_x is the partial image of f at x, that is the image of the function $i \mapsto f(x, i)$.

Relating these notions

These various notions are related one another: classical results show that a finitely computable relation is decidable, that a decidable relation is semi-decidable and that a relation is semi-decidable if and only if it is effectively enumerable.

Indeed, if a relation R is finitely computable, then there exists a function f such that $x\ R\ y$ if and only if y is an element of the finite set $f(x)$. Its characteristic function is therefore computable.

If a relation R is decidable, by composing its characteristic function with a function that maps 1 to 1 and is not defined in 0, we obtain its partial characteristic function, that is therefore computable.

If a relation R is semi-decidable, then there exists a computable function f that takes the value 1 at (x, y) if and only if $x\ R\ y$. We can show that there exists a ternary function g such that $g(x, y, i)$ takes the value 1 if the i first steps of the computation of the value of f at (x, y) have given the result 1 and takes the value 0 otherwise. The binary function h that maps $(x, (y, i))$ to y if $g(x, y, i) = 1$ and an exceptional value otherwise has R_x as partial image at x. The relation R is thus effectively enumerable.

Conversely, if a relation R is effectively enumerable, there exists a function f such that $x\ R\ y$ if and only if there exits an index i such that $y = f(x, i)$. By numbering the elements of the set Ω, we can restrict, without loss of generality, to the case where this set is the set of natural numbers. To determine if two elements x and y are in relation or not, it is sufficient to compute, in a sequential

or parallel way, the values $f(x,0)$, $f(x,1)$, $f(x,2)$, ... and to return the value 1 if the result of one of these computations is equal to y. The partial characteristic function of R is therefore computable.

Notice that, in the proof that if a relation R is semi-decidable then it is effectively enumerable, we can avoid the use of an exceptional value, if all the sets R_x are non empty. In this case, we first show that there exists a computable function d that maps x to an element of R_x. And we define the binary function h as the function that maps $(x,(y,i))$ to y if $g(x,y,i) = 1$ and to $d(x)$ otherwise.

1.4 The notion of non deterministic algorithm

This first approach has lead us to define three sets of relations, as the two last ones coincide. Which one corresponds to the notion of non deterministic algorithm? To answer this question, we must, in a first step, give a precise definition of this notion, that is specify the execution modalities of such an algorithm. One way to do so, is to define a non deterministic algorithm by a not necessarily confluent rewrite system or, equivalently, by a not necessarily deterministic Turing machine.

A rewrite system defined on a language $\mathcal{L}$ containing a special symbol F defines a non deterministic algorithm that maps an element x to all the elements y such that $F(\underline{x})$ reduces to $\underline{y}$, where $\underline{x}$ is the expression of the element x in the language $\mathcal{L}$. For instance, the rewrite system formed with the rules $F(x) \longrightarrow x$ and $F(x) \longrightarrow S(x)$ defines a non deterministic algorithm that maps natural number x to the numbers x and $x+1$. In the same way, the rewrite system formed with the rules $F(x) \longrightarrow x$ and $F(x) \longrightarrow F(S(x))$ defines a non deterministic algorithm that maps a natural number x to all the numbers greater than or equal to x.

It is then easy to prove that the relation defined by such an algorithm is semi-decidable or, equivalently, effectively enumerable, and that, conversely, all such relations can be computed by such a rewrite system. Indeed, if we consider a rewrite system, the function that maps x and i to y if i is a reduction sequence leading from $F(\underline{x})$ to $\underline{y}$, and takes an exceptional value otherwise is computable and its partial image at x is the set of the elements y such that $x\ R\ y$. Conversely, if the relation R is effectively enumerable, there exists a computable function f from $A \times \Omega$ to B such that $x\ R\ y$ if and only if there exists an index i such that $y = f(x,i)$. Numbering the elements of Ω, we can restrict to the case where this set is the set of natural numbers and if the function f is computed by a rewrite system, the relation R is computed by the rewrite system obtained by adding the rules $F(x) \longrightarrow G(x,s)$, $s \longrightarrow 0$, $s \longrightarrow S(s)$ to this system.

The set of relations that captures the notion of non deterministic algorithm is therefore that of the semi-decidable relation or, equivalently, that of the effectively enumerable relations.

1.5 Index, elementary event, possible world, universe and hidden variable

This notion of index, used in the definition of the notion of effectively enumerable set and more generally relation, can be compared to several other notions, used in various domains, that are all concerned, in a way or an other, with the notion of potentiality.

First, it reminds the notion of *elementary event* in Kolmogorov's notion of probability space. The function f_x that maps an index i to a value y could therefore be called a *random variable*, even if the set of indices is not necessarily equipped with a probability distribution. Then, it recalls the notion of *possible world* that is used in Kripke's models of modal logics, and that of *universe* in Everett's interpretation of quantum physics, according to which all the possible results of an experiment, predicted by the theory, indeed happen, but each in a different universe. The fact that $f(x, i) = y$ can therefore be read as the fact that, in the world i, the algorithm f, maps x to y. Finally, it recalls the notion of *hidden variable*, because non-determinism is interpreted as the dependence to an extra variable.

Comparing these different notions, we can remark that various degrees of reality have been given to these indices. In Kripke's models, the possible words are convenient fictions. And there are other types of models of modal logics that are equivalent, but that do not use this notion of possible world. We can probably say the same thing of elementary events in probability theory. In contrast, it seems that Everett and some of his successors have given the same degree of reality to parallel universes as to the different planets of the solar system. Others, in contrast, such as Feynman, seem to have considered them more as convenient fictions. In the same way, the obsolete theories with local hidden variables gave these indices a quite high degree of reality by putting them somewhere in space, this degree of reality is lesser in a theory with non local hidden variables.

The computability theory gives little reality to theses indices because the equivalence of the notions of effectively enumerable and semi-decidable set shows that we can define two equivalent notions by using indices or not.

A question that is very mysterious to me is that of the characterization of the degree of reality given to a notion of index. What does the world "real" means in the sentence "In Everett's interpretation, the parallel universes are real"?

2 Computation over the real numbers

2.1 Computability over the real numbers

We have said that the notion of computability, first defined for functions operating on natural numbers, extends easily to functions operating on countable sets, using a numbering of the elements of these sets. Extending it to functions operating on real numbers is trickier.

A possible definition, that is equivalent to that of Grzegorczyk and Lacombe, as presented by Pour El and Richards [6], is that a function f from an interval I of $\mathbb{R}$ to $\mathbb{R}$ is computable if there exists a computable function e from $\mathbb{Q}^+ \setminus \{0\}$ to $\mathbb{Q}^+ \setminus \{0\}$ and a function F from $\mathbb{Q} \times (\mathbb{Q}^+ \setminus \{0\})$ to $\mathbb{Q}$ such that for all x in I, q in $\mathbb{Q}$ and ε in $\mathbb{Q}^+ \setminus \{0\}$

$$|x - q| \leq e(\varepsilon) \Rightarrow |f(x) - F(q, \varepsilon)| \leq \varepsilon$$

Put in another way, to compute the value of $f(x)$, we must first chose an accuracy ε with which we wish to know the value of $f(x)$. We must then apply the function e to ε to obtain the accuracy with which we are requested to provide the argument x. Then we apply the function F to a rational approximation q of x up to $e(\varepsilon)$ and to ε and we obtained the desired result. By iterating this process, we can obtain approximations of $f(x)$ as accurate as we want, provided that we can supply arbitrarily accurate approximations of x.

However, this definition requires the function e, that maps the desired accuracy ε on the image of f to the requested accuracy on its argument, to be known *a priori* and to be uniform on the interval I. Weihrauch has given an alternative definition that allows to discover this function during the computation. A definition, equivalent to that of Weihrauch, is that a function g from an interval I of $\mathbb{R}$ to $\mathbb{R}$ is computable if there exists a function G from $\mathbb{Q} \times (\mathbb{Q}^+ \setminus \{0\})$ to $\mathbb{Q} \times (\mathbb{Q}^+ \setminus \{0\} \cup \{+\infty\})$ such that for all x in I, q and r in $\mathbb{Q}$, η and ε in $\mathbb{Q}^+ \setminus \{0\}$

$$(G(q, \eta) = (r, \varepsilon) \text{ and } |x - q| \leq \eta) \Rightarrow |g(x) - r| \leq \varepsilon$$

Put in another way, if $(r, \varepsilon) = G(q, \eta)$ and q is an approximation of x up to η, then r is an approximation of $g(x)$ up to ε.

We must however be able to know the value of $g(x)$ with an arbitrary accuracy, which leads to put another condition: if x is a real number and $(q_n)_n$, $(\eta_n)_n$, $(r_n)_n$ and $(\varepsilon_n)_n$ are sequences such that for all n, $(r_n, \varepsilon_n) = G(q_n, \eta_n)$ and $|x - q_n| \leq \eta_n$ and the sequence η goes to 0 at infinity, then the sequence ε also goes to 0 at infinity.

We can then show that if I is a compact interval, that is if it has the form $[a, b]$, then these two definitions are equivalent. These two definitions can generalize to other intervals, in particular to the real line itself. But the second definition generalizes immediately, while the first requires to define first the notion of computable function on a compact interval, and then to use it to define, in second step, the notion of computable function on an arbitrary interval.

Remark that a computable function from an interval I of $\mathbb{R}$ to $\mathbb{R}$ is always continuous on I. Therefore the non continuous functions, for instance the characteristic functions of the sets $\{0\}$, $\mathbb{R}^+$ and $\mathbb{R}^+ \setminus \{0\}$ are not computable.

We have left out of this discussion two other important notions of computability over the real numbers. The first is the notion of *relative* computability, that assumes given a set of non necessarily computable functions, called *oracles*, and study the functions computably definable from these functions. This is, for instance the case of the notion of computability proposed by Blum, Cucker, Schub, and Smale [1], where the characteristic function of equality is

given as an oracle. The second is the notion of non extensional computability, where the value of a function at a real number x may depend not only on this real number x, but also on its presentation, for instance on the Cauchy sequence used to define it. Despite the interest of these notions, in particular in complexity theory for the first and in the theory of constructive proofs for the second, they do not seem to be relevant to the question of the description of physical laws that is discussed here.

Remark that, when extending the notion of computable function to the real numbers, we have not extended the execution modalities of algorithms. We are still using the same execution modalities — rewriting, lambda-calculus, Turing machines, ... Simply, these computation mechanisms are now used with rational numbers approximating the real numbers, using the fact that the set of rational numbers is a countable dense subset of the real line. The fact that the notion of computability over the real numbers is derived from that of computability over the rational numbers is reflected in the fact that all computable functions are continuous.

The thesis that real computable functions are sufficient to describe the laws of nature, or that that an analog machine cannot compute a function that would exceed this notion of computability over the real numbers, implies that all that an analog machine can compute can also be computed by a digital machine, or that a physical process cannot access in a finite time to the infinite amount of information contained in a real number, which can be reformulated in physical terms as the fact that a physical process cannot be used to distinguish, in a finite time, the difference between two infinitely close magnitudes.

2.2 The notion of computable partial function over the real numbers

The second definition, in the style of Weirauch, that we have described above, can easily be generalized to a notion of computable partial function over the real numbers. And this notion will be useful to understand the notion of semi-decidable set of real numbers.

Allowing the function G to be partial is not a very good idea because the computation of $G(q, \eta)$ could terminate, or not, depending on the chosen approximation q of the real x. A better idea is to modify the second condition according to which the sequence ε must go to 0 at infinity. Instead, we require that, for each real number x, either for all sequences $(q_n)_n$, $(\eta_n)_n$, $(r_n)_n$, $(\varepsilon_n)_n$ such that for all n, $(r_n, \varepsilon_n) = G(q_n, \eta_n)$ and $|x - q_n| \leq \eta_n$ and the sequence η goes to 0 at infinity, the sequence ε goes to 0 at infinity, or for all sequences $(q_n)_n$, $(\eta_n)_n$, $(r_n)_n$, $(\varepsilon_n)_n$ such that for all n, $(r_n, \varepsilon_n) = G(q_n, \eta_n)$ and $|x - q_n| \leq \eta_n$ and the sequence η goes to 0 at infinity, the sequence ε is constant and equal to $+\infty$. In this second case, the attempt to obtain any approximation of $g(x)$ by providing more and more accurate approximations of x leads to a computation that does not terminate: the function g is not defined at x.

We can prove this way that the partial characteristic function of the set $\mathbb{R}^+ \setminus \{0\}$ is computable. Indeed, it is computed by the function G that maps

q and η to the ordered pair $(1, \eta)$ if $q - \eta > 0$ and to the ordered pair $(1, +\infty)$ otherwise. Indeed, if x is a strictly positive real, $(q_n)_n$ and $(\eta_n)_n$ two sequences such that for all n, $|x - q_n| \leq \eta_n$ and η goes to 0 at infinity, then the sequence $(q_n - \eta_n)_n$ goes to x at infinity and thus it is strictly positive beyond a certain point. The sequence $(G(q_n, \eta_n))_n$ is therefore equal to $(1, \eta_n)_n$ beyond a certain point and $g(x) = 1$. If x is negative or null, in contrast, we have for all n, $q_n - \eta_n \leq x \leq 0$, thus the sequence $(q_n - \eta_n)_n$ is always negative and the sequence $(G(q_n, \eta_n))_n$ is always equal to $(1, +\infty)$: the function g is not defined in x. We can, this way, express formally what is often stated informally: if x is a strictly positive real, then there is an effective way to prove it.

We can notice that the domain of a partial computable function is always an open set. Indeed, if f is a function and x a real number such that $f(x)$ is defined, then there exist sequences $(q_n)_n$, $(\eta_n)_n$ such that for all n, $|x - q_n| \leq \eta_n$ and the sequence η goes to 0 at infinity. Let $(r_n)_n$ and $(\varepsilon_n)_n$ be sequences such that for all n, $(r_n, \varepsilon_n) = G(q_n, 2\eta_n)$. The function f is defined at x, the sequence $(2\eta_n)_n$ verifies the condition that for all n, $|x - q_n| \leq 2\eta_n$ and it goes to 0 at infinity. Thus, the sequence ε goes to 0 at infinity and there exists a natural number m such that ε_m is different from $+\infty$. The interval $[q_m - 2\eta_m, q_m + 2\eta_m]$ is a neighborhood of x and for all y in this interval, $|y - q_m| \leq 2\eta_m$, thus, there exist two sequences $(q'_n)_n$ and $(\eta'_n)_n$ such that $q'_0 = q_m$ and $\eta'_0 = 2\eta_m$ and for all n, $|y - q'_n| \leq \eta'_n$ and such that the sequence η' goes to 0 at infinity. Let $(r'_n)_n$ and $(\varepsilon'_n)_n$ be the sequences such that for all n, $(r'_n, \varepsilon'_n) = G(q'_n, \eta'_n)$. We have $\varepsilon'_0 = \varepsilon_m$ thus the sequence ε' is not constant and equal to $+\infty$ and f is defined at y.

Thus, if we call *semi-decidable* a set whose partial characteristic function is computable, then all semi-decidable sets of real numbers are open.

3 Non deterministic algorithms over the real numbers

We have extended above the notion of computability of functions operating on natural numbers to relations between natural numbers. We can try to extend, in a similar way, the notion of computability of functions operating on real numbers to relations between real numbers.

As already noticed, in the case of real numbers, we will not be able to refer to execution modalities specific to algorithms operating on real numbers, because, the notion of computability over the real numbers being derived from that of computability over the rational numbers, there is no such execution modalities. Thus, to guide us in the choice of an extension of the notion of computability of functions operating on real numbers to relations between real numbers, we shall need other criteria. We shall require, in particular, that the proposed definition extends the notion of computable function, that is that a functions operating on real numbers is computable as a function if and only if it is computable as a relation.

3.1 The case where the set R_x is finite

Even if the set of finite subsets of the real line has the same cardinality than the real line itself, it is not possible to represent a finite set of real numbers by a real number. This representation is not even possible for two-element sets, because there is no bi-continuous one-to-one mapping between the real line and the part of the plane situated above the diagonal. This jeopardizes the possibility to use the potential finiteness of the set R_x.

In some particular cases, for instance when the cardinal of the set R_x is always the same, or when it is uniformly bounded, we can represent the relation by a tuple of computable functions that map x to the various values the algorithm may take on x. However, this type of representations does not seem to be very general.

3.2 Representing the set R_x as a decidable set

Representing the set R_x as a decidable set leads to represent a relation R by its characteristic function. As we have seen, the computable functions are always continuous, and thus the relations that have a computable characteristic functions are only the empty and the full relation.

This idea also leads to dead end.

3.3 Representing the set R_x as a semi-decidable set

Representing the set R_x as a semi-decidable set leads to represent the relation R by its partial characteristic function. We can, for instance, represent the relation R such that $x\ R\ y$ if $x < y < x + 1$, whose partial characteristic function is $\chi_{\mathbb{R}^+\setminus\{0\}}((x + 1 - y)(y - x))$.

In contrast, the partial characteristic function of the relation R defined by $x\ R\ y$ if $y = x$ or $y = x + 1$ is not computable, because its graph is not an open set. In the same way, the partial characteristic function of the functional relation R defined by $x\ R\ y$ if $y = x$ is not computable.

Thus, if this solution allows more relations to be represented than the previous ones, it is far from being sufficient.

3.4 Representing the set R_x as an effectively enumerable set

Representing the set R_x as an effectively enumerable set leads to represent the relation R by a function f such that $x\ R\ y$ if and only if there exists an index i such that $y = f(x, i)$. Thus, we can define a notion of *effectively enumerable* relation by saying that a relation R is effectively enumerable if there exists a set Ω and a computable function f from $I \times \Omega$ to $\mathbb{R}$ such that for all x, the set R_x is the partial image of f in x, that is the image of the function $i \mapsto f(x, i)$. In this definition, the set Ω may be arbitrary. It can, for instance, be the set

of natural numbers, the real line, an interval of the set of natural numbers, an interval of the real line, ...

This solution, unlike the three previous ones, gives a large set of representable relations, for instance the relation R defined by $x\ R\ y$ if $y = x$ or $y = x + 1$ is representable, taking the set of natural numbers for the set Ω and, for f, the function defined by $f(x, 0) = x$ and $f(x, i) = x + 1$ if $i \geq 1$. This function is computed by the function G where $G(q, \eta, 0) = (q, \eta)$ and $G(q, \eta, i) = (q+1, \eta)$ if $i \geq 1$. The functional relation R defined by $x\ R\ y$ if $y = x$ can also be represented by taking the set of natural numbers for Ω and, for f, the function defined by $f(x, i) = x$, that is computed by the function G defined by $G(q, \eta, i) = (q, \eta)$.

More generally, all computable functions mapping real numbers to the real numbers can be represented as relations, as it suffices to define the function $f(x, i)$ as $g(x)$, ignoring the index i. Conversely, a functional relation that can be represented is computable as a function, because this function associates to x the value $f(x, i_0)$ where i_0 is an arbitrary index.

Remark that, as the characteristic function of equality is not computable, the argument developed in the case of natural numbers that effectively enumerable relations were semi-decidable does not generalize to the case of real numbers. There are many more effectively enumerable relations than semi-decidable ones.

The conclusion of this section is this that among the four sets of relations we have defined, the last one, the set of effectively enumerable relations is the largest. In particular, it is the only one that contains all computable functions and the relation that maps the real x to the real numbers x and $x + 1$, that intuitively seems to be computable. It is the only one that can pretend to be used as a description language for the laws of physics.

4 Probabilities

What happens if we extend this notion of non deterministic algorithm to a notion of probabilistic algorithm?

Trying to represent a non deterministic algorithm as a computable function f mapping an ordered pair (x, y) to the values 1 of 0 depending on the fact that y could be the result of the algorithm at x or not, lead to failure because the function f being continuous, only the empty and full relations could be represented.

We can try, instead, to map the ordered pair (x, y) not to 1 or 0 but to a real number indicating the propension of y to be the result of the algorithm at x. Depending on the cases, this propension can be a probability, a density of probability, it may be also be a number whose square is such a probability or a density of probability, ... This leads us to define a notion of algorithm, not only non deterministic, but also probabilistic.

A probabilistic algorithm, that takes an element of a set A as argument and returns an element of a set B if it terminates, defines a function that maps every element of A to a probability distribution over the set B, that is a function that maps a pair formed with an element x of A and a subset Y of B to the

probability that the value of this algorithm at x is in Y.

Discussing the computability of this function is delicate, because, we have the problem of representing the set Y. We can nevertheless focus on some particular cases. If the set Y is always an interval, we can replace this function by a repartition function, that is a function that maps x and y to the probability that the value of the algorithm at x is less than y. If this repartition function is moreover differentiable, we can replace it by its derivative, that this the function that maps x and y to the density of probability that the value of the algorithm at x is y. Another particular case is when the algorithm can take only a finite or countable number of values for each x. It is then possible to define this probability distribution by a function that maps x and y to the probability that the value of the algorithm at x is exactly y.

If we require this function to be computable, it must be continuous. In the first case, there are many random variables whose repartition function or whose density function is continuous. In the second, the fact that the function that maps x and y to the probability that the value of the algorithm at x is y is continuous implies that if it is different from zero at a point y, it must be minored by a strictly positive value on a neighborhood of y, which is contradictory. Thus, it is not possible to represent this way a probabilistic algorithm that takes two different values at 0 with probabilities $1/2$.

Going from non deterministic algorithms to probabilistic ones thus gives to this method based on characteristic functions a wider spectrum, but not a spectrum wide enough to describe algorithms with a discrete probability distribution. The same argument eliminates also the attempt to base a definition on partial characteristic functions. A partial propension function defined at a point y is defined on a neighborhood of this point and the same problem occurs as in the discrete case.

In contrast, the representation of relations based on the notion of effective enumeration generalizes easily to probabilistic algorithms with discrete probability distributions. We can, for instance, represent the algorithm that maps the real number x to x and $x+1$ with probabilities $1/2$ by taking for the set of indices $\Omega = \{0, 1, 2, 3\}$ and by defining the function f by $f(x, 0) = f(x, 1) = (x, 1/4)$ and $f(x, 2) = f(x, 3) = (x + 1, 1/4)$. If $f(x, i) = (v_i, p_i)$, then the probability that the value of the algorithm at x be y is the sum of the p_i for the i such that $v_i = y$. In this case, the probability that the result of the algorithm at x be x or $x + 1$ is $1/2$ and the probability that the result of the algorithm is another value is 0. This function that takes its value in a cartesian product can be decomposed into two functions h and p such that $h(x, 0) = h(x, 1) = x$, $h(x, 2) = h(x, 3) = x + 1$ and $p(x, 0) = p(x, 1) = p(x, 2) = p(x, 3) = 1/4$. This amounts to define a non deterministic algorithm with the function h and to equip independently the set Ω with a discrete probability distribution.

This solution also permits to represent the algorithm that maps the real number x to x and $2x$ with probabilities $1/2$ and the real number 0 to 0 with probability 1, defining the function f by $f(x, 0) = f(x, 1) = (x, 1/4)$ and $f(x, 2) = f(x, 3) = (2x, 1/4)$. This would not be allowed with a solution that would represent this algorithm as a function mapping each x to the set of pos-

sible results, each with its probability, as it would be discontinuous at 0.

The conclusion of this section is thus that, in the case of probabilistic algorithms, like in the case of non deterministic algorithms, the definition based on effective enumeration is that that permits to represent the largest set of relations. In particular, it is the only one that permits to represent the relation that maps the real number x to x and $x+1$, with probabilities $1/2$, that intuitively seems to be computable.

5 What is a computable description of a non deterministic theory in physics?

This discussion on the nature of non deterministic algorithms on real numbers permits to describe more precisely what it may mean for a theory in physics to have an algorithmic expression or not, when this theory is non deterministic and represents physical magnitudes with real numbers.

To do so, let us imagine an experiment in which one prepares a system by choosing a magnitude x, let the system evolve for a fixed time T, and measures a magnitude y. Our two hypotheses on the theory we try to describe are that it represents magnitudes x and y by real numbers and that for each value x, it does not prescribe a unique result y, but a non empty set of possible results, possibly equipped with probabilities. What would be an algorithm that would describe the results prescribed by the theory, for such an experiment?

As we have seen, we should expect such an algorithm to indicate neither if y is a possible result for the experiment initiated with the value x, nor the probability for this result to be y.

In contrast, the theory should describe this experiment by a function that to each value of x associates a random variable f_x that is itself a function that maps each element of a set of indices Ω to a value y. We can call f the function that maps x and i to the value $f_x(i)$. It is this function, that an algorithmic description of the theory must represent by an algorithm.

We know that there are two possible definitions of the notion of effectively enumerable set, one as the image of a partial computable function and the other as the image of a total computable function. These two definitions are equivalent except for the case of the empty set. In this paper we have favored the second definition, which has lead us to require the function f to be a total computable function. Thus, the algorithm taking the value x and the index i as arguments and computing the value y must always terminate. This is consistent with the intuition that a physical experiment that takes place in a finite time determined *a priori* is described with a terminating algorithm. Only experiments that repeat a measure to decide is the experiment should be continued or terminated may be described by potentially non terminating algorithms.

The non-determinism of a theory means in general that the set of possible results predicted by the theory can have several elements but not that it can have none. We can therefore also state that the algorithm f never fails, as this

possibility of failure has been introduced only to handle the case of the empty set.

We can speculate that nature does not enumerate all the elements i of Ω one after the other to compute the different values of $f(x,i)$ but computes these different values in parallel. This idea may probably be related to the idea of parallelism introduced by Feynman, but the precise articulation of these ideas remains to be explored. We may also speculate that the time necessary to compute the value of $f(x,i)$ is always lower than the time T of the experiment, in other words that the physical time and the logical time are related.

The existence of such an algorithm is a strong constraint on the theory. In all the cases where the experiment is deterministic, it boils down exactly to the deterministic physical Church thesis: the fact that the link between x and y is a computable function. It is therefore neither more constraining nor less constraining for a theory to be algorithmic in the non deterministic case than in the deterministic case.

We shall not answer here to the question of whether such or such theory in physics is algorithmic or not. Our only purpose in this paper was to try to understand how this question could be stated.

Acknowledgments

I want to thank Pablo Arrighi and Jean-Baptiste Joinet for many remarks on a previous draft of this paper, Olivier Bournez, Assia Mahboubi and Nathalie Revol helping me to find my way in the complex domain of real computation and Giuseppe Longo and Thierry Paul for always stimulating discussions.

References

[1] L. Blum, F. Cucker, M. Shub, and S. Smale, *Complexity and Real Computation*, Springer (1998).

[2] G. Dowek, *Les Métamorphoses du Calcul*, Le Pommier (2007).

[3] G. Dowek, The logical analysis of physical phenomena, manuscript (2008).

[4] R. Gandy, Church's Thesis and the Principles of Mechanisms, in J. Barwise, H.J. Keisler, and K. Kunen, *The Kleene Symposium*, North Holland (1980), 123-148.

[5] J.-B. Joinet, Sur le Temps Logique, in J.-B. Joinet *Logique, Dynamique et Cognition*, Publications de la Sorbonne (2007), 31-49.

[6] M.B. Pour-El and J.I. Richards, *Computability in Analysis and Physics*, Springer (1988).

[7] K. Weihrauch, *Computable Analysis, an Introduction*, Springer (1998).