Irreversibility and Complexity

Yair Lapin, The Hebrew University of Jerusalem, Computer Authority Div.

Characteristics of complex systems from the point of view of irreversibility, loss of information[1] and algorithmic complexity

December 2021

Abstract: Complexity is a young science, relatively new and as such there are many aspects of it within the limits of philosophy. This science has gained importance with modern computing that has allowed it to be studied more thoroughly through computational simulations[2], but despite the progress that has been made, there are still characteristics of complex systems that are not very trivial and difficult to understand. Some of these characteristics can be addressed more clearly through information, using simple models of complex systems to which the concept of information can be applied in terms of Shannon for the functions and algorithmic information such as Kolgomorov complexity and logical depth [3]. In computational mathematical models based on recursive processes, recursive functions, and information in functions, it is possible to see aspects of complexity in a different way, its connection with the information lost in the process and irreversibility. The time, the logical depth of the complexity, as the loss of information of this irreversible process. The complexity displayed as a transformation of the input information through the computation of an algorithm.

---

[1] See the concepts of arithmetic logical irreversibility, arithmetic logical entropy, and their interpretation in terms of Shannon information as information loss and uncertainty in my article on the Turing halt problem. Here I extend these concepts to functions in general and applied to complex systems.

[2] S. Wolfram has said in his book NKS that the investigation of complex systems has advanced due to computational models, conventional mathematical models are not enough to understand complex systems that are deeply connected with sophisticated calculations that cannot be performed without powerful computers

[3] Concept of complexity coined by C. Bennet because he did not believe Kolgomorov complexity was sufficient to describe algorithmic complexity because there are programs that take longer to reproduce the same string of data and Bennet thought it appropriate to introduce the time factor that I consider essential here. https://en.wikipedia.org/wiki/Logical_depth

Complex systems and information

Complex systems are characterized by having many components in permanent and deep interaction with each other, and each of them influences one on the other. On the other hand, the interactions between components or parts of the complex system create information. This information created in the links of the interlocking parts is additional information of the system, not visible to the observer, the result of the interactions between the components or parts of the system. This hidden information is considered hidden variables whose ignorance makes impossible an analysis of such a system in a reductionist way because it is not enough to know the functioning of the parts of the system, but it is necessary to know how these parts interact with each other and this hidden information is not easy to observe.

Therefore, the behavior of complex systems cannot be deduced from the analysis of their parts or properties, it is not predictable in a deterministic way. The reductionist models do not serve to treat them, the simplification of these models makes the hidden variables disappear, these variables being fundamental to predict their behavior. Also, the reduction method also does not work because the laws of complex systems are not linear and are not simple, this work addresses the issue of complex systems from information functions and algorithmic information theory. Therefore, I will refer to only 4 important characteristics of complex systems where the information in terms of the concept developed here is clearer and more relevant. This is about visualizing complexity through information theories.

- Feedback Loops and Nested Systems (Recursions)
- Emergence of Information[4]
- Non-linearity (abrupt changes in the system)
- Lack of memory or independence of your history (loss of information)

It seems that irreversibility facilitates complex behaviors and that complex systems are the product of irreversible dynamics.

[4] It is information that arises from the dynamics of the complex system but is not in its separate parts.

These characteristics are treated in this work through simple models in function loops, recursive functions whose loss of information and entropy due to the irreversibility of the functions affects these processes.

Concept of information in functions

In a way like that used to describe information in communications as a stream of symbols passing through a transmission line, information can be defined in functions as the values that are mapped through it to the image of the function. The mapping and the values in the domain of the function that are mapped to, are the function information [5]. In Shannon's information theory each informative event has its probability of appearing, for example, symbols in a message, and the information entropy is related to this probability of the symbols in the message. Something very similar can be done in functions, the informative event can be defined according to the probability of a certain value in the mapping with the number of values in the domain that give this value to the function. The mapping of the function in the image of the function is the distribution of events that have a specific probability according to what values are taken in the definition domain of the function and this is indicated mathematically by means of the pairs (x, y) that is, on the same *y* there can be different *x* in the definition domain such that this changes the probability for said value in the function.

Given a domain of definition to the function F, a finite and discrete set

$$X = \{x_1 x_2 \ldots . x_n\}$$

And an image of this by means of another finite and discrete set

$$Y = \{y_1 y_2 \ldots . y_k\}$$

Therefore

$$\forall x \in X, y \in Y \text{ , } F(x)=y \wedge 1/N(y) = p(y) \leq 1$$

Where N(y) is the number of values in X such that F(x)=y, therefore $N(y)^{-1}$is the probability of the event *y* in the image of the function when there is no condition that

[5] See the description scheme of functions given in the wiki mentioned above: input/function/output, the idea in Shannon's information is basically the same as input/transmitter/output

changes the equiprobability [6]. The pair (x,y) indicates the complete event information of the function when it is known exactly which element of the domain is mapping to *y*. When the function is bijective, a single pair is enough to have the complete information, but when the function is not, several pairs like these will be necessary to have the complete information.

Given these considerations, the Shannon H entropy can be applied to the functions, which would indicate loss of information or uncertainty of y with respect to x.

$$p_i = p(y_i) \wedge F(x_i) = y_i$$

$$H = -\sum_{i}^{k} p_i \log_2 p_i$$

If N(y) > 1, that is, we have more than one value in the domain of F mapping to *y* in the image, the function is not bijective and therefore we cannot know for sure which was the value in the domain that has given a certain value in the image and being the probability is the same in all cases [7]. Therefore, there is an uncertainty about the value that the function mapped to the value in the image and this function is irreversible because of this, then there is a loss of information in the function because it cannot be established with certainty what was the value in the domain that it was mapped to a specific value in the image because there are several possible values.

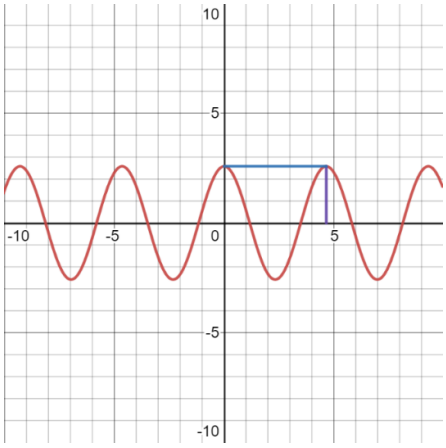
There are many types of functions with many values in the domain that give the same value in the image, thus generating uncertainty regarding the information of these values, for example, we have the functions of trigonometric origin [8]or the logistic function [9]that is an inverted parabola:

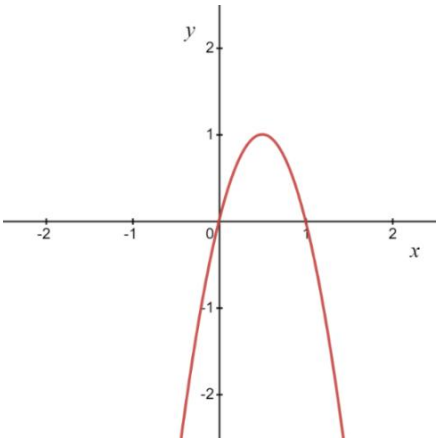
[6] To define the concept of information in functions I stick to equiprobability for simplicity's sake, but perhaps it is also possible to deal with an inequitable probability distribution of domain values.

[7] Perhaps non-equivalent probability distribution can also be applied on these values

[8] $a \cos\left(\frac{2\pi x}{b}\right)$, a=3 and b=4

[9] The logistic mapping function f(x)=rx(1-x) with r=4, see more details in the wiki entry https://en.wikipedia.org/wiki/Logistic_map





Information in different types of functions

There are basically several types of functions for which the concept of information is relevant and important: bijective functions, injective functions and functions that are surjective, but not injective and functions non-surjective that are not injective either. Each type of function has its characteristic Shannon information entropy which indicates loss of information or uncertainty for H>0 or conservation of information for H=0. For example, the isomorphisms that are so important in mathematics, these morphisms being based on bijective functions such that H=0, preserve the information between sets of numbers or objects since the mapping between the domain set and the image set is exact and certain, isomorphisms are informationally conservative.

If a function is bijective, this means that every value of the domain has a value in the image, this is denoted as follows for f:X→Y where X is domain and image:

$$\forall\, a, b \in X\,, f(a) = f(b) \;\rightarrow\; a = b \wedge\; \forall\, y \in Y\, \exists\, x \;\in X \;\rightarrow\; f(x) = y \;\wedge p(y) = 1$$

Then the Shannon entropy of the information for this case would be H=0 because there is no loss of information and the events (x,y) of the function can be determined with certainty.

In the injective function that is not surjective, it has values in the image set that are not defined in the domain of the function

$$\forall\, a, b \in X\,, f(a) = f(b) \;\rightarrow\; a = b \wedge\; \exists\, y \in Y\,, \nexists\; x \;\in X \;\rightarrow\; f(x) = y$$

Therefore, there are domain values that have no image, they are undefined by the function

$$f(y)^{-1} = undef \ \wedge \ N(y) \ > \ 1 \, , p(y) \ = \ \frac{1}{N(y)} \ < \ 1 \rightarrow \ H \ > \ 0$$

That is, it is not possible to determine with certainty the value of the domain that gives us an indefinite in the function, since there are several values for the indefinite in the function, say a subset of the domain, all equally possible or with some probability distribution, an uncertainty is produced on the value not defined in the function and therefore there is a loss of information.

If the function is not injective, in general each value in the image can have more than one value in its domain and this is expressed as follows:

$$\exists \ a, b \in \ X \, , f(a) = f(b) = \ y \wedge a \neq b \ \rightarrow \ p(y) < \ 1 \ \rightarrow \ H \ > \ 0$$

The event (x,y) is not unique and therefore for a given *y* in the image there may be several *x* in the domain and this is what H>0 indicates, there is a loss of information regarding the values in the image and therefore lack of certainty.

An important particular case for recursion is the event (x,undef) , f(x)=undef i.e. when values in the domain are not defined in the function. It may be a subset in the domain of the function that is not defined, if it is the case that N(undef) > 1 and then p(undef)<1, therefore the Shannon entropy H>0. This case is important in recursive processes when it comes to avoiding undefined values that cause recursion failures[10].

Brief introduction to Algorithmic Information Theory[11]

This theory is a combination of Shannon's information theory and Turing's computability theory, it investigates the connection between computability and information. This theory has several areas that have been developed by different people, but what is relevant for this article are the ideas that Chaitin and Kolgomorov developed in parallel.

Shannon developed the idea of information based on statistics, on the probabilities of events or symbols, the work of Chaitin and Kolgomorov added another dimension to this approach, the combinatorial and algorithmic. In the combinatorial plane, the

[10]Many mathematical proofs make use of recursion failures to cause contradictions, these contradictions would basically be the product of the loss of information in the functions if we use this concept.
[11] Also known as Chaitin-Kolgomorov theory

object is treated, not as a statistical distribution of symbols, but as a complex object that has information in a group of objects that also have information, for example a message in a group of messages, which can be combinations of similar messages. In the algorithmic plane Kolgomorov fixed a new concept of complexity, according to this, complex is the opposite of simple or ordered. The less complex object it has less information and the more complex it has more information. This complexity is given by the minimum algorithm that defines the object, that is, this object would be the output of a program, the smallest possible. For example, a low complexity number or message must have a shorter algorithm than its output, i.e., this number or message is producing by a program shorter that their self.

This theory is therefore dedicated to the objects computed by a Turing machine and the information they possess as an output of an algorithm, this object being a string of symbols or rather a binary stream. Every mathematical object can be represented with binary strings, therefore mathematical properties can be investigated by means of this theory, for example, properties of arithmetic that had not been explored before. This theory answers a couple of questions about the nature of binary strings, as outputs of computer programs, their connection to information in Shannon terms, and the relationship of a binary string to randomness. Probability theory did not manage to treat randomness in a profound way according to Kolgonorov, therefore he sought a connection of mathematical objects with much more basic and profound randomness without the need to use statistical distribution or probability but using the concept of algorithm that produces a mathematical object. Chaitin was more interested in the connection of all this with Gödel's incompleteness theorem and Turing's halt problem, he discovered that there are deep connections between Kolgomorov complexity and this theorem and the halt problem.

The Kolgomorov complexity or information content complexity for a string s in a standard Turing machine is defined to be the minimum size of the program S and the shortest input e, which creates as the exact output of this machine. s is said to be compressible by k bits if the program is k bits shorter than s.

The main ideas of algorithmic information theory are as follows:

- A truly random binary string cannot be compressed in an algorithm. Creating a real random binary stream is something very complex and difficult, for a

person it is very difficult to recognize something random because the person tends to look for patterns and these certainly appear in random strings.

- A binary string produced by a program shorter than itself is not random as it can be compressed by the algorithm. Its algorithmic complexity is limited by the entropy of the information of the program that produces it. Kolgomorov proved that no matter what computational language is used for this algorithm, the algorithmic complexity converges to a difference of one constant, therefore it does not depend on which Turing machine or program is used.
- Random strings cannot be built by algorithms or Turing machines.
- There are more random than ordered strings, order is rare, and randomness is common.

One of the most important results of this theory is Chaitin's incompleteness theorem which is closely connected with Gödel's incompleteness theorem and whose result is no less surprising. Chaitin connected the idea of numbers not computable by a Turing machine with the idea of random numbers in Kolgomorov terms.

For example, an irrational number with infinite decimals after the comma as π it has in its digits an approximate probability of $\frac{1}{10}$ and seems to be totally random but it is not algorithmically random because there is a short algorithm that compresses this number, which is very surprising and therefore can be calculated. Chaitin's theorem defined the constant Ω that expresses the probability of a Turing machine stopping its computation with a result, and although this constant is well defined, it cannot be completely calculated, only a part of this number can be calculated. Its characteristics are well known for this constant, but it is not computable and is algorithmically random. Therefore, there is a non-trivial difference between random and algorithmically random, humans can hardly differentiate this, but these concepts would have importance in complex systems, probably one should speak of different levels of complexity as randomness.

The Logical Depth

Logical depth introduces the time factor into Kolgomorov complexity, according to which a logically deep object is produced by a Turing machine very slowly from random input. Deep objects cannot be rapidly produced from shallow

inputs by deterministic or probabilistic processes alike. The fact that an output cannot be produced quickly by an algorithm, that is, by a Turing machine, would mean that it is computable but extremely complex and logically deep. Although a deep object is algorithmically compressible, it has complexity characteristics that cannot be captured by Kolgomorov complexity, and this is the reason why Bennett developed this concept. Deep objects obey the law of slow growth that no fast deterministic algorithm can produce with simple inputs and probabilistic algorithms with perhaps very low probability. Deep objects are usually very difficult to explain by simple causes even though they may be the output of an algorithm, the causal connection would be too unclear to explain them because of the long computations done to produce it and it is also a basically irreversible process because by weakening the causal connection between different parts of the computation, you cannot go back from any step to any other. A simple example of this is a well encrypted message that would take a very long time to find the encryption key if it is not known and this is what most encryption algorithms are based on, brute force it would take thousands of years in the best case. Bidirectional encryption is designed to be able to decrypt the message knowing the key and in a reasonable period of time, that is why the encryptions commonly used in communications are not so logically deep, since that would destroy the message and there would be no way to recover it, but these methods retain certain characteristics of logical depth, since attacking the key if it is unknown is a very long process that has logical depth.

Recursion in mathematics and information

In general, in mathematics and computer science recursion is the call to the same function or method within itself. This form of calling itself is very useful for solving many types of problems, such as efficiently searching for a node in a tree-like graph, for example, or certain calculations such as the factorial of a number. But it is a very expensive method from the point of view of resources because the intermediate values must be saved to make the final calculation when the halt condition is met or, in the mentioned example, return to the root of the tree, which is the reference point for all search in a tree. The halt condition is a final case that if it is not well defined the computation does not stop until the resources are exhausted or it stops with no result because an intermediate result in a certain step is not defined for the function and that causes an irremediable failure in the recursion. Recursive

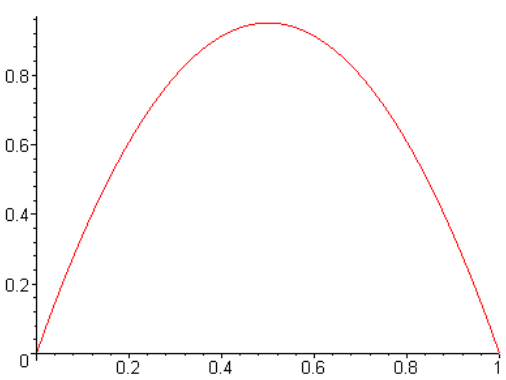
functions, that is, functions that call themselves, are very sensitive to failures for this reason.

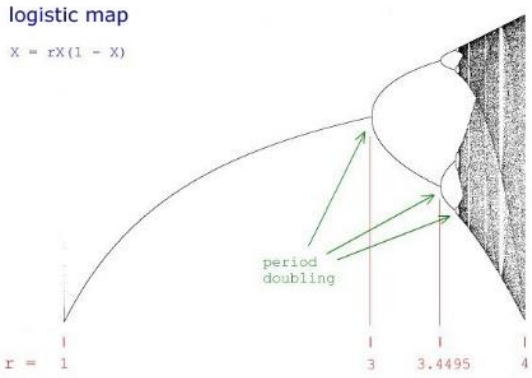
The power of recursion is in being able to define an infinite set of objects or values by finite means because recursion can define infinite computations in a very simple and finite way. But here is the problem when we mess with infinity using finite information, if the stop condition is not met, the computation never ends and in that case the recursion is not very useful since some result is expected in a reasonably finite time. That is why in recursion it is very important to define the final case well and set the halt condition correctly and know for sure that this condition is going to be met with certainty or at least with high security.

Recursion describes an infinite stream of information, being able to receive a finite part of it by limiting the number of recursive loops, but theoretically for this to work in infinity, recursion must lose information and only retain what is necessary so as not to exhaust resources [12]. When such a stream is random, each new bit of information is more important than when the stream is ordered, and this is in accordance with Shannon's theory. For example, [13] the recursion of the logistic function which is a well-known example of dynamical chaos with bifurcations for the attractors[14] of the output/input values when r>3.5

$$x_{n+1} = r.x_n(1 - x_n) \ \wedge \ x \ \in \ [0,1)$$

[12] A server-type program is a good example, infinite loops and keeps only what is necessary to perform the required computations, the data that you want to keep is saved in a database or file, the rest is eliminated.
[13] The graph of the logistic function for r=4 is given above, the larger r, the steeper the inverted parabola, which seems to produce large jumps in the recursion.
[14] Attractors are lines of values to which series mapping outputs converge between oscillations, in logistic mapping exists these bifurcations of convergences, the outputs tend to converge to several attractors in parallel. And these bifurcations are connected to the parameter r, the larger r is, there more attractors 2,4,8... The logistic mapping shows characteristics of the fractals because the bifurcation diagrams are self-similar as can be seen in figure b.





(a) (b)

In (a) we see the logistic mapping parabola while (b) shows its chaotic behavior when the value of r increases, producing important jumps in the outputs that will be new inputs and these outputs converge to several parallel values called attractors.

But to better analyze this example it will be necessary to give certain important characteristics of chaos and its connection with complex systems.

The chaos

Complexity theory lies between determinism and randomness which is complex, and this is known as the edge of chaos. Chaos in general shows very complex information, information in a more extreme situation than lack of order, but according to algorithmic complexity the most complete randomness cannot be produced by any algorithm, although there is chaos produced by quite simple algorithms which has been very surprising. Therefore, there are different types of chaos, it is said that there are hierarchies of chaos, there would be chaos compressible by algorithms and those that are not.

Chaos in general is indeterministic and unpredictable, a chaotic system has no memory, is independent of its history and is very sensitive to small changes. The accumulation of irreversible and unpredictable events creates these chaotic situations. Chaotic systems are a subset of complex systems. Logistic recursion is a very good example of chaos generated by an irreversible function and for that reason it is a good model to visualize the effect of irreversibility and entropy of functions in a complex system such as logistic recursion, but it is not completely

random according with algorithmic information because its complexity is bounded by an algorithm.

Due to its chaotic behavior, it can be said that each bit that the logistic mapping function [15]loses has more importance than in an ordered stream of values that can perhaps be compressed and therefore part of those bits can be dispensed with [16]. The information of functions applied to this case would be related to the distribution of logistic recursion outputs that for each output *y* there are two possible values *x* and *-x* that can give this same output value when this input has been output from other possible values. Another characteristic of this recursion is that small input changes produce sudden changes in the output that will be the next input of the next step, and this generates unpredictable, chaotic behavior. It also has bifurcations, that is, for the same x there can be several values and something that further complicates the information in this recursion. This example has several characteristics of complex systems: feedback loops, high sensitivity to the input data and independence from its history since it is impossible to determine which input gave the next output in a deterministic way, one input can give by 2 different outputs. A small program of this algorithm it can generate pseudo-random numbers and was thus used in the past by computers, but because they are pseudo-random, they can be compressed more than into a completely random series of numbers. They would not be logically deep either because the calculations are not that long, but still there is chaos.

Recursive Functions and Church's λ Calculus[17]

Recursion is also very important in the mathematical definition of effective automatic computation. The effective computation is well defined by Church's λ calculus [18]. The λ calculus is a formal logical system based on functions, recursion, and variable substitution. This is considered the minimal programming language because it defines very simple rules for variable substitution and its use in recursive

[15] f(x)=rx(1-x)
[16] That's what compression means, you can get rid of part of the information bits without losing their content, but compression is very connected to randomness according to algorithmic information theory, randomness is not compressible every bit is important.
[17] Church's λ calculation, the recursive functions in this concept are not the recursion mentioned above
[18] More details in the wiki entry https://en.wikipedia.org/wiki/Lambda_calculus

functions [19]. It is well proven that the calculus λ is Turing complete, that is, it is equivalent to a Turing machine, but it is closer to software than to hardware. Its importance lies in the fact that it can be expressed through this, any algorithm that can be done in a Turing machine through recursive functions and substitution of variables by specific values.

On the other hand, if the recursive functions that describe an algorithm are irreversible and therefore lose information according to the previous definitions, in general the whole process loses information, and this would mean that part of the information is preserved to continue the calculation and it's transformed by the algorithm, and another part is forgotten, this makes the calculation itself irreversible because without the lost information the previous state cannot be recovered. The information that passes through the different stages of the computation would be partial and therefore the total information that would be involved in the computation would be much greater than its total result. The raw information in an algorithm given an input to the program would be greater than the information in the result. The computation would work like to a steam or combustion engine in which part of the energy does work and part is dissipated or lost as heat [20].

Conclusions

It is well known that the behavior of an algorithm cannot be known based on its computational mathematical description, even if it is very simple, such as the recursion of the logistic function already mentioned before, whose behavior is a known example of chaos with bifurcations for parameters that make very steep to the parabola and therefore small changes in the domain cause large changes in the image that only the display of outputs through computation shows its true complexity. The Kolgomorov complexity of the logistic algorithm is not very high, since the size of the program that describes it in any programming language is quite short, but this program displays a very high complexity, creating a stream of semi-random numbers. As well as the mentioned algorithm for calculating the number π, this would be very short compared to the information displayed in π, as is known, any

[19] This concept is different than the recursion in functions mentioned before.

[20] Baez's work on algorithmic thermodynamics basically deals with this idea, but from a thermodynamic point of view without connections to Shannon information

possible sequence of numbers can be found in the decimals of this number, the problem is where the searched sequence occurs. There are numerous examples of the amazing fact that a very simple and short algorithm can display a great complexity of images, a world of complex landscapes like fractals [21]that have very simple rules of production, but its deployment gives very complex images or digital worlds created with small programs [22]. Many of these algorithms have significant logical depth because they cannot run quickly, a large computing power and a long time are necessary to calculate them, therefore their causes are not as trivial as it is believed. The complexity of these outputs would be emerging information, which is not in the algorithm itself, but arises from it due to its logical depth and sophisticated calculations that lose information in the computation[23].

This phenomenon has also been studied by S. Wolfram in cellular automata[24] whose rules are very simple but the complexity that they display in real time is completely unpredictable, a phenomenon that he has called computational irreducibility[25] because it is necessary to run the program and display its outputs for a long time to appreciate their complexity. In the cellular automata that Wolfram has thoroughly investigated, some are more interesting and complex than others, but it is not possible to know in advance, sometimes it even takes time to discover special patterns because their causes are not trivial, they are logically deep.

All this leads me to think that there is a deep connection between logical depth, irreversibility, and loss of information in the computational process. Deep processes lose more information than shallow processes and their reversibility is much less likely.

---

[21] Many complex fractals consume many computational resources and time, which is why it was necessary for very powerful computers to appear in order to know their total deployment.

[22] There is an interesting paper by John Baez on thermodynamics and computation, also on Kolgomorov complexity and Chaitin incompleteness, https://math.ucr.edu/home/baez/thermo/ . Baez expreses similares ideas.

[23] Also, Wolfram in his NKS emphasizes the fact that very simple programs produce very complex outputs and goes further by suggesting that it is not necessary to add randomness to the programs, often the complexity arises only from the sophisticated calculations of the algorithm.

[24] According to the categories given by Wofram, I am referring to class 3 pseudo-random and type 4 computationally irreducible automata, since not all cellular automata have chaotic or complex behavior, many are very predictable.

[25] It is a concept that goes beyond the predictability of an algorithm and is related to the problem of undecidability in computation.

The mathematical description of the algorithm or its programming code are not convincing enough to show the logical depth given by the display of their outputs, its scope cannot be described with all its complexity, the code is a potential, a macro[26] description , but the deployment is at a micro level and can be complex, not trivial, hence its unpredictable and undecidable dynamics, the question of what this or that algorithm will do based on its macro description. Apparently, there is an information difference between the macro level of description and the information at a micro level due to the irreversibility of the process in complex systems, the information entropy Shannon can perhaps describe this difference, the gap for those levels of description. It is known that to study complex systems, its description scope must be well defined because there is a gap between the different levels of description that are not compatible due to the loss of information between these levels. Kolgomorov complexity could describe another dimension of this, the behaviors between the algorithmic macro and the non-algorithmic, algorithmic object versus non-algorithmic object, that is, the complex that can be known and what seems to be beyond our reach.

Bibliography

1. John Baez and Mike Stay, Algorithmic thermodynamics , *Math. Struct. Comp. Sci.* **22** (2012), 771–787.
2. A. Ben-Naim, Information Theory, Part I: An Introduction to Fundamental Concepts, World Scientific, 2017
3. Bennett, Charles H. (1988), "Logical Depth and Physical Complexity", in Herken, Rolf (ed.), *The Universal Turing Machine: A Half-Century Survey,* Oxford U. Press, pp. 227–257
4. Yair Lapin, Arithmetic logical Irreversibility, and the Turing's Halt Problem, Academia, 2021
5. James Gleick, Chaos – The Making of a Science, Critique 2012
6. James Gleick, The Information, Critique 2012
7. S. Wolfram, New Kid of Science, 2002

[26] This would be computational irreducibility too

8. Nigel Goldenfeld & Leo Kadanoff, Simple Lessons from Complexity, Science, April 1999, Vol 28