An Algorithmic Refinement of Maxent Induces a Thermodynamic-like Behaviour in the Reprogrammability of Generative Mechanisms

Hector Zenil*[1,2,3], Narsis A. Kiani[1,2,3], and Jesper Tegnér[2,4]

[1]Algorithmic Dynamics Lab
[2]Unit of Computational Medicine, Centre for Molecular Medicine, Science For Life Laboratory (SciLifeLab), Department of Medicine Solna, Karolinska Institute, Stockholm, Sweden.
[3]Algorithmic Nature Group, LABORES, Paris, France
[4]Biological and Environmental Sciences and Engineering Division,, Computer, Electrical and Mathematical Sciences and Engineering, Division, King Abdullah University of Science and, Technology (KAUST), Kingdom of Saudi Arabia.

Abstract

Reprogramming a generative mechanism to produce a different object is associated with a cost. Here we use the notion of algorithmic randomness to quantify such a cost when reprogramming networks. We identify an asymmetry in a measure of reprogrammability, suggesting an analogy with a thermodynamic asymmetry. The principle of maximum entropy (Maxent) quantifies the evolution of entropy or the uncertainty during state transitions in systems confined to an equilibrium condition. Here we define a generalisation based on algorithmic randomness not restricted to equilibrium physics, based on both distance to algorithmic randomness and reprogrammability. We advance a constructive preferential attachment algorithm approximating a maximally algorithmic random network. Hence, as a refinement on classical Maxent, networks can be quantified with respect to their distance to a maximally algorithmic random network. Our analysis suggests that the reprogrammability asymmetry originates from its non-monotonic relationship to algorithmic randomness. Our analysis motivates further work on the degree of algorithmic asymmetries in systems depending on their reprogrammability capabilities.

Keywords: second law of thermodynamics; reprogrammability; algorithmic complexity; generative mechanisms; deterministic systems; algorithmic randomness; principle of maximum entropy; Maxent.

*Corresponding author: hector.zenil [at] algorithmicnaturelab [dot] org
Invited contribution to *The interplay of thermodynamics and computation in both natural and artificial systems* to be published by the Santa Fe Institute and edited by David Wolpert, Chris Kempes, Joshua Grochow and Peter Stadler https://www.santafe.edu/research/projects/thermodynamics-computation

1 Classical Thermodynamics and related work

According to Arnold Sommerfield, quoted in Califano's seminal book on thermodynamics [5], the business of Entropy and thermodynamics is incredibly messy and misunderstood even by researchers in the area:

> Thermodynamics is a funny subject. The first time you go through it, you don't understand it at all. The second time you go through it, you think you understand it, except for one or two small points. The third time you go through it, you know you don't understand it, but by that time you are so used to it, it doesn't bother you any more.

Conventionally, and traditionally, agnostic thermodynamic Entropy is defined as follows:

- a measure of *statistical* disorder;
- some quantity or property that increases but never decreases;
- a process that defines the direction of time;
- a measure of *statistical* information

Some of the problems surrounding the characterisations of the second law and Entropy go back about a hundred years, when they were introduced, and while most of the discussion around thermodynamics is not only legitimate but central to the most pressing and important questions in physics, the statistical version of Entropy has gained renewed application in areas as diverse as such typicality analysis in the form of the so-called *Principle of Maximum Entropy*, often denoted by *Maxent*.

Previous work has considered the question of replacing part or all of the statistical machinery from statistical mechanics to arrive at an algorithmic approach to thermodynamics. Some authors have discussed the analogy between algorithms and entropy. Perhaps the first example is the thought experiment 'Maxwell's demon', in which Maxwell suggested that the second law of thermodynamics might hypothetically be violated by introducing intelligence in the form of a being capable of following an algorithm that enabled it to distinguish and choose between particles of high and low energy–taken together with Szilard's discussion of this paradox [7]. More recent examples combining computation and entropy can be found in Seth Lloyd's concept of *thermodynamic depth* [12], heavily indebted to the work of Kolmogorov and Chaitin and to Bennett's notion of *logical depth* [3]; in Baez's *algorithmic thermodynamics* approach that is capable of defining an algorithmic version of *algorithmic temperature* and *algorithmic pressure*, and in Crutchfield's *computational mechanics* using *epsilon-machines* [6].

The interest in introducing algorithmic complexity to questions of thermodynamics and the second law derives from the standpoint of introducing an additional dimension in the analysis. For example, testing or refining the second law of thermodynamics by taking into consideration and ruling out apparent disordered states that are not algorithmically random. Unlike statistical

mechanical approaches, algorithmic complexity represents a generalisation over Entropy that assigns lower Entropy (and thus higher causal content) to objects that not only appear statistically simple but are also algorithmically simple by virtue of having a short generating mechanism capable of reproducing the causal content of a network. Without such an additional dimension, causal and non-causal networks are collapsed into the same typical Bernoulli distribution of Entropy in which maximal Entropy represents apparent statistical disorder, without distinguishing between networks with a causal origin and actual random states. Indeed, a random-looking system with maximal Entropy can be recursively generated by a short computer program that Entropy would classify as statistically random but not algorithmically random. An example of a fundamental limitation of Shannon Entropy is, for example, offered in [27], where its fragility is exposed in a very simple example when trying to quantify the deterministic vs random nature of exactly the same object (a recursive graph of algorithmic randomness).

While we will show that the mathematics of changes in algorithmic complexity and the reprogramming capabilities of computer programs show a thermodynamic-like phenomenon that is similar, if not equivalent, to the mathematics of physical thermodynamics, in this paper we do not aim to connect physical thermodynamics to algorithmic thermodynamics directly. In other words, while we may be able to count the number of operations to convert one computer program into another by targeting a specific desired output from those programs, here we do not enter into the details of the energetic cost of implementing these operations in hardware. We believe, however, that these results, while abstract in nature, have a fundamental character reminiscent of the guiding principles of classical thermodynamics.

2 Thermodynamics of computer programming

A thermodynamic-like result can be found in a measure of *sophistication* based on quantifying the difficulty of reprogramming an object or system depending on its initial conditions. A measure of sophistication is a measure capable of telling apart 'sophisticated' cases from simple and random objects, the latter being assigned low complexity. Another measure of this kind is Bennett's *logical depth* [3] (or LD).

2.1 Measures of reprogrammability

Measures of reprogrammability (denoted by $Pr(G)$ and $PA(G)$ introduced in [26] can differentiate between elements that may move an object or a system G towards randomness $\sigma_N(G)$ or away from randomness $\sigma_P(G)$ as follows:

- Relative (re)programmability: $Pr(G) := MAD(\sigma(G)))/n$ or 0 if $n = 0$, where $n = \max\{|\sigma(G)|\}$ measures the shape of $\sigma_P(G)$ and how it deviates from other distributions (e.g. uniform or normal).

- Absolute (re)programmability: $PA(G) := |S(\sigma_P(G)) - S(\sigma_N(G))|/m$, where $m := \max\{S(\sigma_P(G)), S(\sigma_N(G))\}$, where $m = \max(S(\sigma_P(G)), S(\sigma_N(G)))$ and S is an interpolation function. This measure of reprogrammability captures not only the shape of $\sigma_P(G)$ but also the sign of $\sigma_P(G)$ above and below $x = 0$.

For a complete graph all nodes and all edges should have the same algorithmic-information contribution, and thus $\sigma(G)$ can be analytically derived (a flat uniform distribution x = $\log_2 |V(G)|$ with $|V(G)|$ the node count of G). Thus all the nodes of a complete graph are 'slightly' positive (or more precisely, neutral, if they are 'positive' by only $\log_2 |V(G)|$).

```
(1)Print[1] 200 times                (2) Print[s]
1111111111111111111                  00110010010010110011
1111111111111111111                  01100101110010011111
1111111111111111111                  00110101000110011101
1111111111111111111                  11001010100010011100
1111111111111111111                  10000101000011011100
1111111111111111111          s =     11000101000110010100
1111111111111111111                  11010111000110110011
1111111111111111111                  01111100000001010110
1111111111111111111                  00010110110011010111
1111111111111111111                  01011010100010111101
```

Figure 1: Thermodynamic reprogrammability. Top: The programs producing simple versus random data have different reprogrammability properties. If repurposed to generate programs to print blocks of 0s, we only need a single intervention in the generative program of (1), changing 1 to 0 inside the **Print** instruction indicating that 200 0s be printed instead of 1s. In contrast, asking a program that prints a random binary string s to print only 0s will require on average $|s|/2$ interventions to manually change every bit 1 to 0. Random perturbations can be seen as the exploration of the possible paths through which an object may evolve over time. Thus uniform random perturbations provide a picture of the set of possible future states. This means that, in both cases, the asymmetric cost of moving random to simple and simple to random from a purely algorithmic perspective is also relevant in the case of naturally evolving systems.

Another way to illustrate the phenomenon of asymmetric reprogrammability is by considering networks as computer programs (Fig. 2) (or as produced by computer programs). The algorithmic complexity K of a complete graph k grows by its number of nodes because the generating mechanism is of the form "connect all N nodes", where $N = |V(k)|$. In contrast, the algorithmic complexity of a random Erdös-Rényi (E-R) graph with edge density ~ 0.5 grows by the number of edges $|E(\text{E-R})|$ because to reproduce a random graph from scratch the sender would need to specify every edge connection, as there is no

way to compress the description.

As depicted in 2, removing a node n from a complete graph k produces another complete, albeit smaller graph. Thus the generating program for both k and $k' = k\backslash n$ is also the relationship between k and k', which is causal. In contrast, if an edge e is removed from k, the generating program of $k'' = k\backslash e$ requires the specification of e and the resulting generating program of $C(k'') > C(k)$, sending k towards randomness for every e randomly removed.

On the one hand, moving a complete graph towards a random graph (see Fig. 1 and 2) requires random changes if we are only interested in reproducing the statistical properties of the random graph, that is, its degree distribution, requiring no previous knowledge. On the other hand, we see how a random graph can also be easily rewired towards a complete graph by simply adding the edges needed to make it complete. However, if the complete graph is required to exactly reproduce a specific random graph and not just its statistical properties, then one would need to have full knowledge of the specific random graph and apply specific changes to the complete graph, making the process slow and requiring a lot of knowledge. Yet, moving the random graph to the complete graph still requires the same effort as before, because the complete graph is unique, given the fixed number of nodes. Nevertheless, moving a simple graph, such as the complete graph (see Fig. 2) by edge removal has a greater impact on its algorithmic complexity than performing the same operation on a random graph. Specifically, if S is a simple graph and R a random one, then we have it that $C(S) - C(S\backslash e) > C(R) - C(R\backslash e)$, i.e. the rate change from S to (a non-specific) R is greater than in the other direction, thus imposing a thermodynamic-like asymmetry related to the difficulty in reprogramming one object into another according to its initial program-size complexity. The asymmetric axis where the highest reprogrammability point can be found is exactly the point at which $C(S) - C(S\backslash e) = C(R) - C(R\backslash e)$ for a specific S and R.

It is clear then how analysing the contribution of each element to the object, as shown in Fig. 2, has the potential to reveal the algorithmic nature of the original object and how difficult it is to reprogram the underlying generative computer program in order to produce a different output/graph.

A thermodynamic-like effect can be found in the (re)programmability capabilities of an object. Moving random networks by edge removal is significantly more difficult than moving simple networks towards randomness. For random graphs, there are only a few elements, if any, that can be used to move then slowly towards simplicity, as shown in Fig. 2. In contrast, a larger number of elements can move a simple network faster towards randomness. This relationship, captured by the reprogrammability rate for S simple versus R random graphs, induces a thermodynamic-like asymmetry based on algorithmic complexity and reprogrammability.

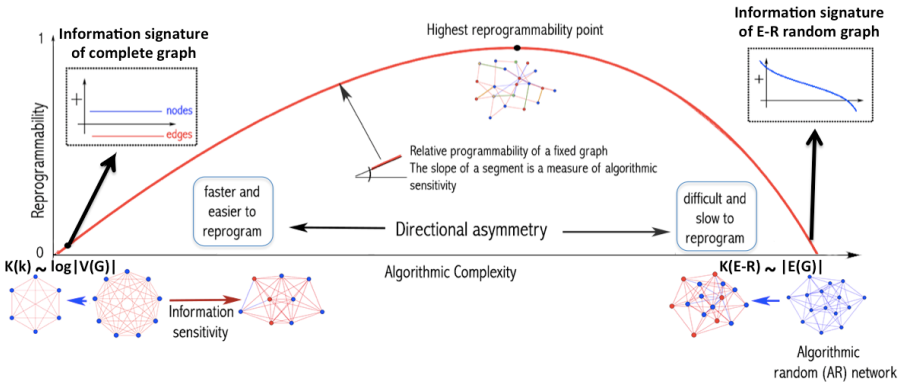

Figure 2: Networks as programs. All real-world networks lie between the extreme cases of being as simple as a complete graph whose algorithmic complexity K is minimal and grows by only $\log |V(k)|$, or a random (also statistically random and thus E-R) graph whose algorithmic complexity is maximal and grows by its number of edges $|E(\text{E-R})|$. If we ask what it takes to change the program producing k to produce E-R and vice versa, in a random graph, any single node or edge removal does not entail a major change in the program-size of its generating program, which is similar in size to the random graph itself i.e. $|E(G)|$. The curve shows how, without loss of generality, the reprogramming capability of networks as computer programs produces an asymmetry imposed by algorithmic complexity and reminiscent of traditional thermodynamics as based on classical probability. A maximally random network has only positive (blue) elements (Fig. 5) because there exists no perturbation that can increase the randomness of the network either by removing a node or an edge, as it is already random (and thus non-deterministic). Thus changing its (near) minimal program-size length by edge or node removal is slow. However, a simple graph may have elements that push its program-size length towards randomness. In each extreme case (simple vs random) the distribution of sorted elements capable of pushing in each direction is shown in the form of what we call 'signatures', both for edge and node removal. The highest reprogrammability point is the place where a graph has as many elements to push it in one direction as in the other.

3 Algorithmic information dynamics

At a fundamental level, Shannon Entropy and algorithmic complexity are very similar. Indeed, the expected value of algorithmic entropy equals its Shannon entropy up to a constant that depends only on the distribution [1]. That is, every deterministic source has both low Entropy and low algorithmic randomness, and algorithmically random objects surely have the highest Shannon Entropy. However, in practical terms, they are fundamentally different. Nowhere in Shannon Entropy is there any indication as to how to estimate the underlying mass

probability distribution needed to determine the random or deterministic nature of a source, the availability of some other method for doing so being simply assumed. Algorithmic complexity, however, does provide many methods, albeit very difficult ones, to estimate the algorithmic randomness of an object by looking at the set of possible programs of at most the size of the program that may produce the object. One popular way to approximate it has been by using lossless compression algorithms, given that a compressed program is sufficient proof of non-randomness.

The algorithmic (Kolmogorov-Chaitin) complexity of a string x is the length of the shortest effective description of x. There are several versions of this notion. Here we use mainly the plain complexity, denoted by $C(x)$.

We work over the binary alphabet $\{0,1\}$. A string is an element of $\{0,1\}^*$. If x is a string, $|x|$ denotes its length. Let M be a universal Turing machine that takes two input strings and outputs one string. For any strings x and y, we define the algorithmic complexity of x conditioned by y with respect to M, as:

$$C_M(x|y) = \min\{|p| \textit{ such that } M(p,y) = x\}.$$

We match the machine M with a universal machine U, thereby allowing us to drop the subscript. We then write $C(x|y)$ instead of $C_M(x|y)$. We will also write $C(x)$ instead of $C(x|\lambda)$ (where λ is the empty string).

3.1 A calculus of algorithmic change

At the core of the reprogrammability analysis is a causal calculus as introduced in [26] based upon the change in complexity of a system subject to perturbations, particularly the direction (sign) and magnitude of the change in algorithmic information content C between two states of the same object, such as objects G and G', which for purposes of illustration can be graphs with a set of nodes $V(G)$ and a set of edges $E(G)$.

The dynamics of a graph can then be defined as transitions between different states, and one can always ask after the potential causal relationship between G and G'. In other words, what possible underlying minimal computer program can explain G' as evolving over discrete time from state G?

For graphs, we can allow the operation of edge e removal from G denoted by $G\backslash e$ where the difference $|C(G) - C(G\backslash e)|$ is an estimation of the shared algorithmic mutual information of G and $G\backslash e$ or the *algorithmic information dynamics* (or *algorithmic dynamics* in short) for evolving time-dependent systems (e.g. if G' evolves from G after t steps). If e does not contribute to the description of G, then $|C(G) - C(G\backslash e)| \sim \log_2 |V(G)|$, where $|V(G)|$ is the node count of G, i.e. the algorithmic dynamic difference will be very small and at most a function of the graph size, and thus the relationship between G and G' can be said to be causal and not random, as G' can be derived from G' with at most $\log_2$ bits. If, however, $|C(G) - C(G\backslash e)| > \log_2 |V(G)|$ bits, then G and $G\backslash e$ do not share causal information, and the removal of e results in a loss. In

contrast, if $C(G) - C(G\backslash e) > n$, then e cannot be explained by G alone nor is it algorithmically not contained/derived from G, and it is therefore a fundamental part of the description of G with e as a generative causal mechanism in G, or else it is not part of G but has to be explained independently, e.g. as noise. Whether it is noise or part of the generating mechanism of G depends on the relative magnitude of n with respect to $C(G)$ and on the original causal content of G itself. If G is random, then the effect of e will be small in either case, but if G is richly causal and has a very small generating program, then e as noise will have a greater impact on G than would removing e from the description of an already short description of G. However, if $|C(G) - C(G\backslash e)| \leq \log_2 |V(G)|$, where $|V(G)|$ is the vertex count of G, then e is contained in the algorithmic description of G and can be recovered from G itself (e.g. by running the program from a previous step until it produces G with e from $G\backslash e$).

For example, in a complete graph (see Fig. 2), the removal of any single node leads to a logarithmic reduction in its algorithmic complexity, but the removal of any single edge leads to an increase in randomness. The former because the result is simply another complete graph of a smaller size, and the latter because the deleted link would need to be described after the description of the complete graph itself.

If a graph is evolving deterministically over time, its algorithmic complexity remains (almost) constant up to a logarithmic term as a function of time because its generating mechanism is still the same, but if its evolution is non-deterministic and possible changes to its elements are assumed to be uniformly distributed then a full perturbation analysis can simulate their next state, basically applying exhaustively all possible changes one step at a time. In this case, any node/edge perturbation to a simple graph has a very different effect than performing the same interventions to a random graph.

4 Principle of maximum algorithmic randomness

There is a wide range of applications in science, in particular in statistical mechanics, as ways to study and model the typicality of cases and objects. Indeed, determining how removed an object is from maximum Entropy has been believed to be an indication of its typicality based on its information content.

Maximum entropy, or simply Maxent, is taken as the state of a system when it is at its most statistically disordered—when it is supposed to encode the least information.

We have seen here, however, that Entropy may collapse cases that are only random-looking but are not truly so. Based on the ideas above we can, nevertheless, suggest a conceptual and numerical refinement of classical Maxent by an algorithmic Maxent. Instead of making comparisons against a maximal Entropy graph, by generating a MAR graph one can replace such a comparison by using the MAR graph (shown in red in Fig. 3) by either comparing the original graph

(denoted by G in Fig. 3) or a compressed version (denoted by $minG$) approximatedby, for example, algorithmic graph sparsification [24]. Such comparisons are marked as t and t' in Fig. 3 and replace the need to make comparison with a maximal Entropy graph that may not be algorithmic random. In Fig. 3 it is shown how such a replacement can be made and what comparisons are now algorithmic (t and t') versus only statistical, which in the case of this illustration shows a graph that has been shown to produce a near maximally degree sequence when it is of lowest algorithmic random [27] (in magenta, top right).

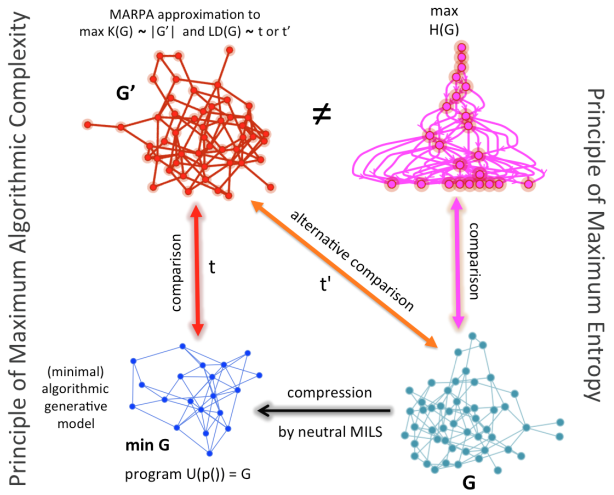

Figure 3: The paths to statistical and algorithmic randomness are different, and they determine different principles for different purposes. Classical Maxent quantifies statistical randomness but its algorithmic refinement quantifies both statistical and algorithmic randomness. This opens up the range of possibilities for moving toward and reaching a random graph, by not only considering whether it is random-looking but also whether it is actually algorithmically random.

4.1 Maximal Algorithmic Randomness Preferential Attachment (MARPA) algorithm

Once the number of nodes is fixed, a MAR graph is of density 0.5, just like a classical E-R graph. This is because the highest algorithmic complexity is reached when $K(G) \sim |E(G)|$ is maximised exactly between the 2 extreme cases in which fully disconnected and complete graphs reach minimum complexity $K(G) \sim |V(G)|$.

This means that, without loss of generalisation to other objects, a Maximal Algorithmic Random graph G is an Erdös -Rényi (E-R) graph that is algorithmically random, i.e. whose shortest possible computer description is not

(much) shorter than $|E(G)|$, where $|E(G)|$ is the number of edges of G; or, $|E(G)| - C(G) < c$.

MARPA seeks to maximise the information content of a graph G by adding new edges (or nodes) at every step. The process approximates a network of a given size that has the largest possible algorithmic randomness and is also an Erdös-Rényi (ER) graph. An approximation of a 'Maximal' Algorithmic-Random (MAR) graph can be produced as a reference object whose generating program is not smaller than the network (data) itself and can better serve in maximum (algorithmic-) entropy modelling.

MARPA allows constructions of a maximally random graph (or any object) by edge preferential attachment, in such a manner that randomness increases for any given graph. Let G be a network and $C(G\backslash e)$ the information value of e with respect to G such that $C(G) - C(G\backslash e) = n$. Let $P = \{p_1, p2, \ldots, p_n\}$ be the set of all possible perturbations. P is finite and bounded by $P < 2|E(G)|$ where $E(G)$ is the set of all elements of G, e.g. all edges of a network G. We can find the set of perturbations e' in P such that $C(G) - C(G\backslash e') = n'$ with $n' < n$. As we iterate over all e in G and apply the perturbations that make $n' < n$, for all e, we go through all $2^{|E(G)|}$ possible perturbations (one can start with all $|E(G)|$ single perturbations only) maximising the complexity of $G' = max\{G|C(G) - C(G\backslash e) = \{\text{max among all } p \text{ in } P \text{ and } e \in G\}$.

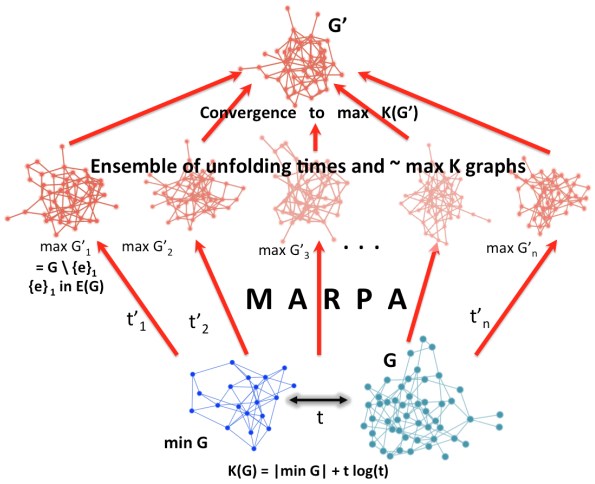

Figure 4: While many objects can look statistically random, there are less that are actually algorithmic random and finding approximations to it will converge given that algorithmic complexity is upper semi-computable meaning one can approximate it from above hence tighter bounds for a MAR graph can always be found with greater computational power. This algorithmic version expands and improves over classical Maxent which is based classical information theory and over Shannon Entropy.

The purpose of MARPA (see Fig. 4) is to estimate the algorithmic com-

plexity of an object by traditional means such as popular lossless compression algorithms (e.g. LZW), which are limited but are an improvement over Entropy [18] and go in the algorithmic direction, or more sophisticated means can be used as introduced in in [10, 15, 20, 25, 23] based on algorithmic probability.

Alternatively, there is a configuration of all edges in G that maximises the algorithmic randomness of G. Let such a maximal complexity be denoted by $maxC(G)$. Then we find the sequential set of perturbations $\{P\}$ such that $maxC(G) - C(G) = 0$, where $C(G) - maxC(G)$ is a measure, related to randomness deficiency, of how removed G is from its (algorithmic-)randomized version $maxC(G)$ (notice that $C(G)$ is upper bounded by $maxC(G)$, and so the difference is always positive).

Theorem 1. *(Existence) There is an E-R graph that is not algorithmically random.*

Proof. We can build an E-R graph by using a pseudo-random generator such that the edge density of the graph is exactly 0.5, and having all the required statistical properties found in any typical E-R graph, yet being by definition of low algorithmic complexity because a pseudo-random generator is a computer program of fixed length, exhibiting an E-R graph that is not algorithmically random. □

Corollary 2. *(Not uniqueness) There is more than one E-R graph that is not algorithmically random.*

It follows then that an E-R graph with density 0.5 may be of maximal entropy, but of high or low algorithmic randomness, i.e. either recursively generated or not.

One can also consider the absolute maximum algorithmic-random graph, that we will denote by $maxC(G)$. $maxC(G)$ is a graph comprising the same number of nodes but with an edge rearrangement operation such that $C(G) < C(max(G)) \leq 2^k$, where $k = (|E(G)|(|E(G)|-1))/4$ is the maximum number of edges in G divided by 2 where at edge density 0.5 it reaches maximal algorithmic randomness. The process of pushing a given network thus approximates a target network of a size that has the greatest possible algorithmic randomness and is also an Erdös-Rényi (E-R) graph. The pseudo-code is as follows:

1. Start with graph G.
2. Perform on G to produce G' such that $C(G') > C(G)$.
3. $G := G \cup e_{j'}$
4. Repeat 1 until final target size and quality of algorithmic randomness desired (e.g. by rate divergence from a statistical random graph).

Operations to produce G' can be of many types, edge rearrangement, ede addition or edge removal.

Notice that the algorithm can also start from empty, for which one would be generating approximations to all MAR graphs (for that operation) of all sizes from the bottom up.

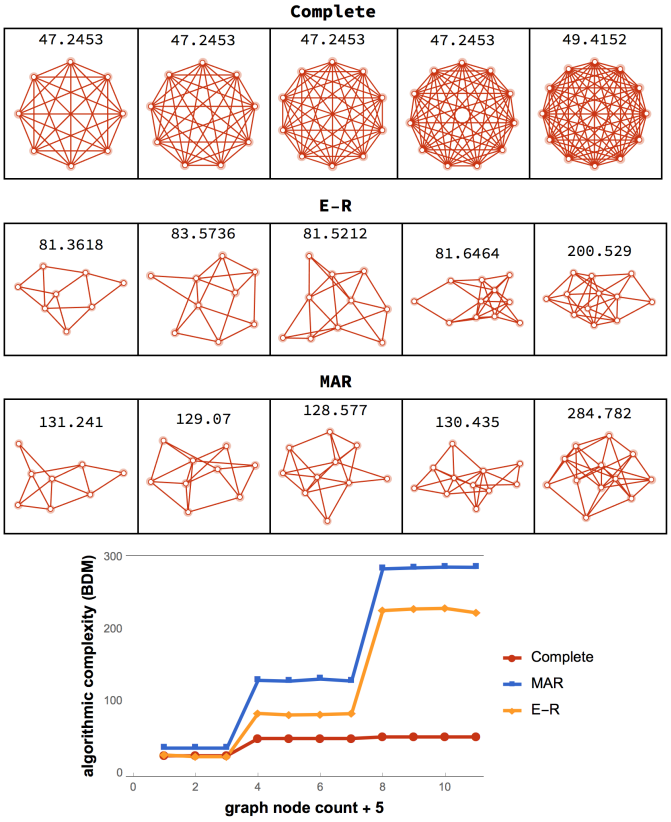

Figure 5: Generating approximations to small MAR graphs by using methods introduced in [19, 20, 25, 23]. The progression of the complexity estimations (top) indicates the progression. For complete graphs, their estimation grows logarithmic as theoretically predicted. Between E-R and MAR graphs a difference is shown for the same number of nodes and edges with their values diverging (see also Fig. 5) numerically showing that not all E-R graphs are also the highest algorithmic random as they can be identified to have been produced by short generative programs. Bottom plot: Estimations of the complexity of a complete graph, for any method, including Shannon Entropy and popular lossless compression algorithms (such as LZW, Compress) will only grow logarithmically as predicted by the theory, here depicted are values of complexity as approximated by methods based on algorithmic probability as introduced in [20, **?**, 25, 23]. In contrast, E-R are of maximal Entropy by definition yet they are not necessarily algorithmic random. The greatest approximations to maximal algorithmic random graphs (MAR) are shown for the same number of nodes than the complete and E-R graphs. The jumps in the plot are explained by a bounded and convergent error coming from a partition function used in the implementation of the estimating method and fully explained in [25]. The trend of the functions are, however, not only clear but in line with theoretical expectations.

Approximating a MAR candidate is computationally very expensive, with exponential time complexity in $O(2^{n^2})$ because at every step all possible attachments have to be tested and evaluated (i.e. all possible permutations of the adjacency matrix of size $n \times n$). But small MAR graphs and minor but fundamental improvements over only random-looking graphs, are computationally feasible.

Unlike some E-R graphs, MAR graphs cannot, in principle, be generated by computer programs of much smaller size than the edge count of the networks themselves. The intuition behind the construction of a MAR graph is that the shortest computer program (measured in bits) that can produce the adjacency matrix of a MAR graph, is of about the size of the adjacency matrix and not significantly shorter.

4.2 Numerical examples

Using methods based on algorithmic probability as introduced in [20, 19] and [15, 25, 23] moves simple and complex networks towards and away from randomness. The reversed history of the structure found in a random graph provides an alternative path for moving networks and graphs towards randomness. The structure found indicates what must be deleted to push an E-R graph towards a maximally algorithmic random (MAR) graph.

In Fig. 5 it is shown how the complexity of graphs with exactly the same number of nodes and edges that comply with the properties of an E-R graph (e.g. edge density 0.5) do not comply with the property of maximum algorithmic randomness and their values will diverge for typical randomly chosen examples of growing graphs by node count. It is also shown how the numerical approximations, both for simple (complete) and MAR graphs follow the theoretical expectations.

5 Discussion and conclusion

We have established parallels between computer programming as based on the dynamics of computer programs to be repurposed and emphasising a thermodynamic-like phenomena when considering rewiring along an axis defined by algorithmic randomness. The lack of symmetry in the operation of rewiring networks with regards to algorithmic randomness implies a direction or an asymmetry. We have also introduced a principle akin of maximum entropy but based on algorithmic complexity. The principle of maximum algorithmic randomness can therefore be viewed as a refinement of Maxent together with algorithms to estimate algorithmic random graphs, including some small cases which we numerically calculated and included in this study.

Acknowledgements

H.Z. wishes to acknowledge the support of Swedish Research Council (Vetenskapsrådet) grant No. 2015-05299.

References

[1] A. Teixeira, A. Matos, A. Souto, L. Antunes, *Entropy* 13 (3), pp. 595–611, 2011.

[2] J.C. Baez, M. Stay, Algorithmic Thermodynamics, Computability of the Physical, *Mathematical Structures in Computer Science*, 22, 771–787, 2012.

[3] C.H. Bennett, Logical Depth and Physical Complexity, in R. Herken, *The Universal Turing Machine: a Half-Century Survey*, Oxford University Press, pp. 227-257, 1988.

[4] T. Böttcher, An Additive Definition of Molecular Complexity, *J. Chem. Inf. Model.*, 2016, 56 (3), pp 462-470

[5] S. Califano, *Pathways to Modern Chemical Physics*, Springer, Springer-Verlag Berlin Heidelberg, 2012.

[6] J.P. Crutchfield and C.R. Shalizi. Thermodynamic depth of causal states: Objective complexity via minimal representations, *Phys. Rev. E*, 59(1):275-283, 1999.

[7] L. Szilard, On the decrease of entropy in a thermodynamic system by the intervention of intelligent beings, Zeit. Phys. 53, pp. 840–856, 1929 (English translation in H. S. Leff and A. F. Rex (eds.) Maxwell?s Demon. Entropy, Information, Computing, Adam Hilger, Bristol, 1990.

[8] G.J. Chaitin. On the length of programs for computing finite binary sequences *Journal of the ACM*, 13(4):547–569, 1966.

[9] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, 2nd Edition, Wiley-Blackwell, 2009.

[10] J.-P. Delahaye and H. Zenil, Numerical Evaluation of the Complexity of Short Strings: A Glance Into the Innermost Structure of Algorithmic Randomness, *Applied Mathematics and Computation* 219, pp. 63–77, 2012.

[11] A. N. Kolmogorov. Three approaches to the quantitative definition of information, *Problems of Information and Transmission*, 1(1):1–7, 1965.

[12] S. Lloyd, H. Pagels, Complexity as thermodynamic depth, *Annals of Physics*, 188, pp. 186–213, 1988.

[13] P. Martin-Löf, The definition of random sequences, *Information and Control*, 9:602–619, 1966.

[14] F. Soler-Toscano, H. Zenil, A Computable Measure of Algorithmic Probability by Finite Approximations with an Application to Integer Sequences, Complexity vol. 2017, 2017.

[15] F. Soler-Toscano, H. Zenil, J.-P. Delahaye and N. Gauvrit, *Calculating Kolmogorov Complexity from the Frequency Output Distributions of Small Turing Machines*, PLoS ONE 9(5): e96223, 2014.

[16] M. Spevack, The Harvard Concordance to Shakespeare. *The Belknap Press of Harvard University Press*, Cambridg. Mass., 1973.

[17] R.J. Solomonoff, A formal theory of inductive inference: Parts 1 and 2. *Information and Control*, 7:1–22 and 224–254, 1964.

[18] H. Zenil, L. Badillo, S. Hernández-Orozco and F. Hernández-Quiroz, Coding-theorem Like Behaviour and Emergence of the Universal Distribution from Resource-bounded Algorithmic Probability, *International Journal of Parallel Emergent and Distributed Systems*, 2018 DOI: 10.1080/17445760.2018.1448932

[19] H. Zenil, F. Soler-Toscano, K. Dingle and A. Louis, Graph Automorphisms and Topological Characterization of Complex Networks by Algorithmic Information Content, Physica A: Statistical Mechanics and its Applications, vol. 404, pp. 341–358, 2014.

[20] H. Zenil, F. Soler-Toscano, J.-P. Delahaye and N. Gauvrit, *Two-Dimensional Kolmogorov Complexity and Validation of the Coding Theorem Method by Compressibility*, 2013.

[21] H. Zenil, N.A. Kiani and J. Tegnér, Algorithmic complexity of motifs clusters superfamilies of networks, Proceedings of the *IEEE International Conference on Bioinformatics and Biomedicine,* Shanghai, China, 2013.

[22] L. Levin, Laws of information conservation (non-growth) and aspects of the foundation of probability theory, *Problems in Form. Transmission* 10. 206–210, 1974.

[23] H. Zenil, N.A. Kiani and J. Tegnér, Methods of Information Theory and Algorithmic Complexity for Network Biology, *Seminars in Cell and Developmental Biology*, vol. 51, pp. 32-43, 2016.

[24] H. Zenil, N.A. Kiani, J. Tegnér, Unsupervised and Universal Data Reduction and Network Sparsification Methods By Minimal Algorithmic Information Loss, arXiv:1802.05843.

[25] H. Zenil, F. Soler-Toscano, N.A. Kiani, S. Hernández-Orozco, A. Rueda-Toicen, A Decomposition Method for Global Evaluation of Shannon Entropy and Local Estimations of Algorithmic Complexity, arXiv:1609.00110 [cs.IT].

[26] H. Zenil, N.A. Kiani, F. Marabita, Y. Deng, S. Elias, A. Schmidt, G. Ball, J. Tegnér, An Algorithmic Information Calculus for Causal Discovery and Reprogramming Systems, bioRxiv 185637; doi: `https://doi.org/10.1101/185637`.

[27] H. Zenil, N.A. Kiani and Jesper Tegnér, Low Algorithmic Complexity Entropy-deceiving Graphs, *Phys. Rev. E.*, 96, 012308, 2017.