Hindawi
Complexity
Volume 2017, Article ID 3250301, 12 pages
https://doi.org/10.1155/2017/3250301

Research Article

On Measuring the Complexity of Networks: Kolmogorov Complexity versus Entropy

Mikołaj Morzy,[1] Tomasz Kajdanowicz,[2] and Przemysław Kazienko[2]

[1] *Institute of Computing Science, Poznan University of Technology, Piotrowo 2, 60-965 Poznań, Poland*
[2] *Department of Computational Intelligence, ENGINE-The European Centre for Data Science, Wrocław University of Science and Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland*

Correspondence should be addressed to Mikołaj Morzy; mikolaj.morzy@put.poznan.pl

Received 6 April 2017; Revised 27 July 2017; Accepted 13 August 2017; Published 1 November 2017

Academic Editor: Pasquale De Meo

One of the most popular methods of estimating the complexity of networks is to measure the entropy of network invariants, such as adjacency matrices or degree sequences. Unfortunately, entropy and all entropy-based information-theoretic measures have several vulnerabilities. These measures neither are independent of a particular representation of the network nor can capture the properties of the generative process, which produces the network. Instead, we advocate the use of the algorithmic entropy as the basis for complexity definition for networks. Algorithmic entropy (also known as Kolmogorov complexity or K-complexity for short) evaluates the complexity of the description required for a lossless recreation of the network. This measure is not affected by a particular choice of network features and it does not depend on the method of network representation. We perform experiments on Shannon entropy and K-complexity for gradually evolving networks. The results of these experiments point to K-complexity as the more robust and reliable measure of network complexity. The original contribution of the paper includes the introduction of several new entropy-deceiving networks and the empirical comparison of entropy and K-complexity as fundamental quantities for constructing complexity measures for networks.

1. Introduction

Networks are becoming increasingly more important in contemporary information science due to the fact that they provide a holistic model for representing many real-world phenomena. The abundance of data on interactions within complex systems allows network science to describe, model, simulate, and predict behaviors and states of such complex systems. It is thus important to characterize networks in terms of their complexity, in order to adjust analytical methods to particular networks. The measure of network complexity is essential for numerous applications. For instance, the level of network complexity can determine the course of various processes happening within the network, such as information diffusion, failure propagation, actions related to control, or resilience preservation. Network complexity has been successfully used to investigate the structure of software libraries [1], to compute the properties of chemical structures [2], to assess the quality of business processes [3–5], and to provide general characterizations of networks [6, 7].

Complex networks are ubiquitous in many areas of science, such as mathematics, biology, chemistry, systems engineering, physics, sociology, and computer science, to name a few. Yet the very notion of network complexity lacks a strict and agreed-upon definition. In general, a network is considered "complex" if it exhibits many features such as small diameter, high clustering coefficient, anticorrelation of node degrees, presence of network motifs, and modularity structures [8]. These features are common in real-world networks, but they rarely appear in artificial random networks. Finding a good metric with which one can estimate the complexity of a network is not a trivial task. A good complexity measure should not depend solely on the number of vertices and edges, but it must take into consideration topological characteristics of the network. In addition, complexity is not synonymous with randomness or unexpectedness. As has been pointed out [8], within the spectrum of possible networks, from the most ordered (cliques, paths, and stars) to the most disordered (random networks), complex networks occupy the very center of this spectrum. Finally, a good complexity measure should not depend on a particular network representation and should yield consistent results for various representations of the same network (adjacency matrix, Laplacian matrix, and degree sequence). Unfortunately, as current research suggests, finding a good complexity measure applicable to a wide variety of networks is very challenging [9–11].

Among many possible measures which can be used to define the complexity of networks, the entropy of various network invariants has been by far the most popular choice. Network invariants considered for defining entropy-based complexity measures include number of vertices, number of neighbors, number of neighbors at a given distance [12], distance between vertices [13], energy of network matrices such as Randić matrix [14] or Laplacian matrix [15], and degree sequences. There are multiple definitions of entropies, usually broadly categorized into three families: thermodynamic entropies, statistical entropies, and information-theoretic entropies. In the field of computer science, information-theoretic measures are the most prevalent, and they include Shannon entropy [16], Kolmogorov-Sinai entropy [17], and Rényi entropy [18]. These entropies are based on the concept of the information content of a system and they measure the amount of information required to transmit the description of an object. The underlying assumption of using information-theoretic definitions of entropy is that uncertainty (as measured by entropy) is a nondecreasing function of the amount of available information. In other words, systems in which little information is available are characterized by low entropy and therefore are considered to be "simple." The first idea to use entropy to quantify the complexity of networks comes from Mowshowitz [19].

Despite the ubiquitousness of general-purpose entropy definitions, many researchers have developed specialized entropy definitions aimed at describing the structure of networks [10]. Notable examples of such definitions include the proposal by Ji et al. to measure the unexpectedness of a particular network by comparing it to the number of possible network configurations available for a given set of parameters [20]. This concept is clearly inspired by algorithmic entropy, which defines the complexity of a system not in terms of its information content, but in terms of its generative process. A different approach to measure the entropy of networks has been introduced by Dehmer under the form of information functional [21]. Information functional can be also used to quantify network entropy in terms of k-neighborhoods of vertices [12, 13] or independent sets of vertices [22]. Yet another approach to network entropy has been proposed by Körner, who advocates the use of stable sets of vertices as the basis to compute network entropy [23]. Several comprehensive surveys of network entropy applications are also available [9, 11].

Within the realm of information science, the complexity of a system is most often associated with the number of possible interactions between elements of the system. Complex systems evolve over time, they are sensitive to even minor perturbations at the initial steps of development and often involve nontrivial relationships between constituent elements. Systems exhibiting high degree of interconnectedness in their structure and/or behavior are commonly thought to be difficult to describe and predict, and, as a consequence, such systems are considered to be "complex." Another possible interpretation of the term "complex" relates to the size of the system. In the case of networks, one might consider to use the number of vertices and edges to estimate the complexity of a network. However, the size of the network is not a good indicator of its complexity, because networks which have well-defined structures and behaviors are, in general, computationally simple.

In this work, we do not introduce a new complexity measure or propose new informational functional and network invariants, on which an entropy-based complexity measure could be defined. Rather, we follow the observations formulated in [24] and we present the criticism of the entropy as the guiding principle of complexity measure construction. Thus, we do not use any specific formal definition of complexity, but we provide additional arguments why entropy may be easily deceived when trying to evaluate the complexity of a network. Our main hypothesis is that algorithmic entropy, also known as Kolmogorov complexity, is superior to traditional Shannon entropy due to the fact that algorithmic entropy is more robust, less dependent on the network representation, and better aligned with intuitive human understanding of complexity.

The organization of the paper is the following. In Section 2, we introduce basic definitions related to entropy and we formulate arguments against the use of entropy as the complexity measure of networks. Section 2.3 presents several examples of entropy-deceiving networks, which provide both motivation and anecdotal evidence for our hypothesis. In Section 3, we introduce Kolmogorov complexity and we show how this measure can be applied to networks, despite its high computational cost. The results of the experimental comparison of entropy and Kolmogorov complexity are presented in Section 4. The paper concludes in Section 5 with a brief summary and future work agenda.

2. Entropy as the Measure of Network Complexity

2.1. Basic Definitions. Let us introduce basic definitions and notation used throughout the remainder of this paper. A *network* is an ordered pair $G = \langle V, E\rangle$, where $V = \{v_1, \ldots, v_N\}$ is the set of *vertices* and $E = \{(v_i, v_j) \in V \times V\}$ is the set of *edges*. The *degree* $d(v_i)$ of the vertex v_i is the number of vertices adjacent to it, $d(v_i) = |\{v_j : (v_i, v_j) \in E\}|$. A given network can be represented in many ways, for instance, using an *adjacency matrix* defined as

$$A_{N\times N}[i,j] = \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

An alternative to the adjacency matrix is the Laplacian matrix of the network defined as

$$L_{N\times N}[i,j] = \begin{cases} d(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j,\ (v_i, v_j) \in E \\ 0 & \text{otherwise.} \end{cases} \qquad (2)$$

Other popular representations of networks include the *degree list* defined as $D = \langle d(v_1), d(v_2), \ldots, d(v_n)\rangle$ and the *degree distribution* defined as

$$P(d_i) = p(d(v_j) = d_i) = \frac{\left|\{v_j \in V : d(v_j) = d_i\}\right|}{N}. \qquad (3)$$

Although there are numerous different definitions of entropy, in this work we are focusing on the definition most commonly used in information sciences, the Shannon entropy [16]. This measure represents the amount of information required to provide the statistical description of the network. Given any discrete random variable X with n possible outcomes, the Shannon entropy $H(X)$ of the variable X is defined as the function of the probability p of all outcomes of X:

$$H(X) = -\sum_{i=1}^{n} p(x_i)\log_b p(x_i). \qquad (4)$$

Depending on the selected base of the logarithm, the entropy is expressed in bits ($b = 2$), nats ($b = e$), or dits ($b = 10$) (bits are also known as Shannon, and dits are also known as Hartley). The above definition applies to discrete random variables; for random variables with continuous probability distributions differential entropy is used, usually along with the limiting density of discrete points. Given a variable X with n possible discrete outcomes such that in the limit $n \to \infty$ the density of X approaches the invariant measure $m(x)$, the continuous entropy is given by

$$\lim_{n\to\infty} H(X) = -\int p(x)\frac{p(x)}{m(x)}\,dx. \qquad (5)$$

In this work, we are interested in measuring the entropy of various network invariants. These invariants can be regarded as discrete random variables with the number of possible outcomes bound by the size of the available alphabet, either binary (in the case of adjacency matrices) or decimal (in the case of other invariants). Consider the 3-regular graph presented in Figure 1. This graph can be described using the following adjacency matrix:

$$A_{10\times 10} = \begin{bmatrix} 0&0&1&0&1&0&0&0&1&0 \\ 0&0&0&0&1&0&0&1&1&0 \\ 1&0&0&1&0&1&0&0&0&0 \\ 0&0&1&0&0&1&1&0&0&0 \\ 1&1&0&0&0&0&0&0&0&1 \\ 0&0&1&1&0&0&0&1&0&0 \\ 0&0&0&1&0&0&0&1&0&1 \\ 0&1&0&0&0&1&1&0&0&0 \\ 1&1&0&0&0&0&0&0&0&1 \\ 0&0&0&0&1&0&1&0&1&0 \end{bmatrix}. \qquad (6)$$

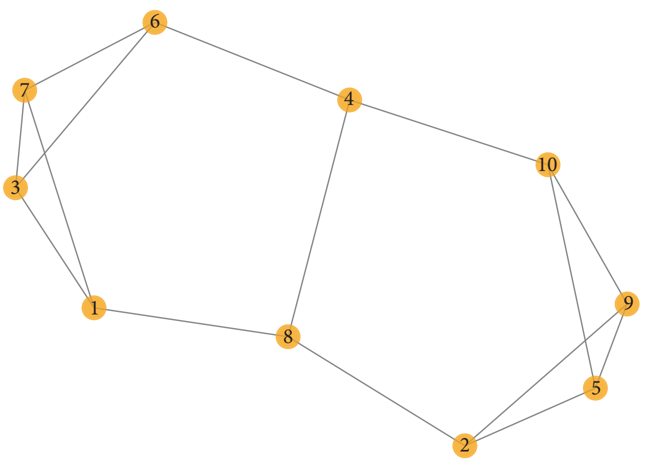


Figure 1: Three-regular graph with 10 vertices.

This matrix, in turn, can be flattened to a vector (either row-wise or column-wise), and this vector can be treated as a random variable with two possible outcomes, 0 and 1. Counting the number of occurrences of these outcomes, we arrive at the random variable $X = \{x_0 = 0.7, x_1 = 0.3\}$ and its entropy $H(X) = 0.88$. Alternatively, this graph can be described using the degree list $D = \langle 3,3,3,3,3,3,3,3,3,3\rangle$ which can be treated as the random variable with the entropy $H(D) = 0$. Yet another possible random variable that can be derived from this graph is the degree distribution $PD = \{d_0 = 0, d_1 = 0, d_2 = 0, d_3 = 1\}$ with the entropy $H(PD) = 0$. In summary, any network invariant can be used to extract a random variable and compute its entropy.

Thus, in the remainder of the paper, whenever mentioning entropy, we will refer to the entropy of a discrete random variable. In general, the higher the randomness, the greater the entropy. The value of entropy is maximal for a random variable with the uniform distribution and the minimum value of entropy is attained by a constant random variable. This kind of entropy will be further explored in this paper in order to reveal its weaknesses.

As an alternative to Shannon entropy, we advocate the use of Kolmogorov complexity. We postpone the discussion of Kolmogorov complexity to Section 3, where we provide both its definition and the method to approximate this incomputable measure. For the sake of brevity, in the remainder of this paper, we will use the term "entropy" to refer to Shannon entropy and the term "K-complexity" to refer to Kolmogorov complexity.

2.2. Why Is Entropy a Bad Measure of Network Complexity.

Zenil et al. [24] argue that entropy is not appropriate to measure the true complexity of a network and they present several examples of networks which should not qualify as complex (using the colloquial understanding of the term), yet which attain maximum entropy of various network invariants. We follow the line of argumentation of Zenil et al., and we present more examples of entropy-deceiving networks. Our main aim is to show that it is relatively easy

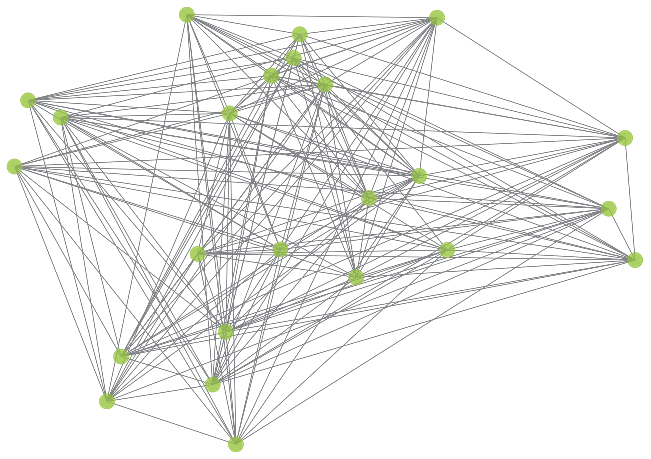


Figure 2: Block network composed of eight of the same 3-node blocks.

to construct a network which achieves high values of entropy of various network invariants. Examples presented in this section outline the main problem with using entropy as the basis for complexity measure construction: namely, that entropy is not aligned with intuitive human understanding of complexity. Statistical randomness, as measured by entropy, does not imply complexity in a useful, operational way.

The main reason why entropy and other entropy-related information-theoretic measures fail to correctly describe the complexity of a network is the fact that these measures are not independent of the network representation. As a matter of fact, this remark applies equally to all computable measures of network complexity. It is quite easy to present examples of two equivalent lossless descriptions of the same network having very different entropy values, as we will show in Section 2.3. In this paper, we experiment with four different representations of networks: adjacency matrices, Laplacian matrices, degree lists, and degree distributions. We show empirically that the choice of a particular representation of the network strongly influences the resulting entropy estimation.

Another property which makes entropy a questionable measure of network complexity is the fact that entropy cannot be applied to several network features at the same time, but it operates on a single feature, for example, degree and betweenness. In theory, one could devise a function which would be a composition of individual features, but high complexity of the composition does not imply high complexity of all its components and vice versa. This requirement to select a particular feature and compute its probability distribution disqualifies entropy as a universal and independent measure of complexity.

In addition, an often forgotten aspect of entropy is the fact that measuring entropy requires making an arbitrary choice regarding the aggregation level of the variable, for which entropy is computed. Consider the network presented in Figure 2. At the first glance, this network seems to be fairly random. The density of the network is 0.35 and its entropy computed over adjacency matrix is 0.92 bits. However, this network has been generated using a very simple procedure. We begin with the initial matrix:

$$M_{3\times 3} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}. \tag{7}$$

Next, we create 64 copies of this matrix, and each of these copies is randomly transposed. Finally, we bind all these matrices together to form a square matrix $M_{24\times 24}$ and we use it as the adjacency matrix to create the network. So, if we were to coalesce the adjacency matrix into 3×3 blocks, the entropy of the adjacency matrix would be 0, since all constituent blocks are the same. It would mean that the network is actually deterministic and its complexity is minimal. On the other hand, it should be noted that this shortcoming of entropy can be circumvented by using the *entropy rate* (n-gram entropy) instead, because entropy rate calculates the entropy for all possible levels of granularity of a variable. Given a random variable $X = \langle x_1, x_2, \dots, x_n \rangle$, let $p(x_i, x_{i+1}, \dots, x_{i+l})$ denote the joint probability over l consecutive values of X. Entropy rate $H_l(X)$ of a sequence of l consecutive values of X is defined as

$$H_l(X) = -\sum_{x_1\in X} \cdots \sum_{x_l\in X} p(x_1,\dots,x_l)\log_2 p(x_1,\dots,x_l). \tag{8}$$

Entropy rate of the variable X is simply the limit of the above estimation for $l \to \infty$.

2.3. Entropy-Deceiving Networks.

In this section, we present four different examples of entropy-deceiving networks, similar to the idea coined in [24]. Each of these networks has a simple generative procedure and should not (intuitively) be treated as complex. However, if the entropy was used to construct a complexity measure, these networks would have been qualified as complex. The examples given in this section disregard any specific definition of complexity; their aim is to outline main shortcomings of entropy as the basis for any complexity measure construction.

2.3.1. Degree Sequence Network.

Degree sequence network is an example of a network which has an interesting property: there are exactly two vertices for each degree value $1, 2, \dots, N/2$; $N = |V|$.

The procedure to generate degree sequence network is very simple. First, we create a linked list of all N vertices, for which $d(v_1) = d(v_N) = 1$ and $\forall i \neq 1, i \neq N, d(v_i) = 2$. It is a circle without one edge (v_1, v_N). Next, starting with vertex v_3, we follow a simple rule:

```
for $i = 3$ to $N/2$ do
  for $j = 1$ to $(i - 2)$ do
    add_edge($v_i, v_{N/2+j}$)
  end for
end for
```

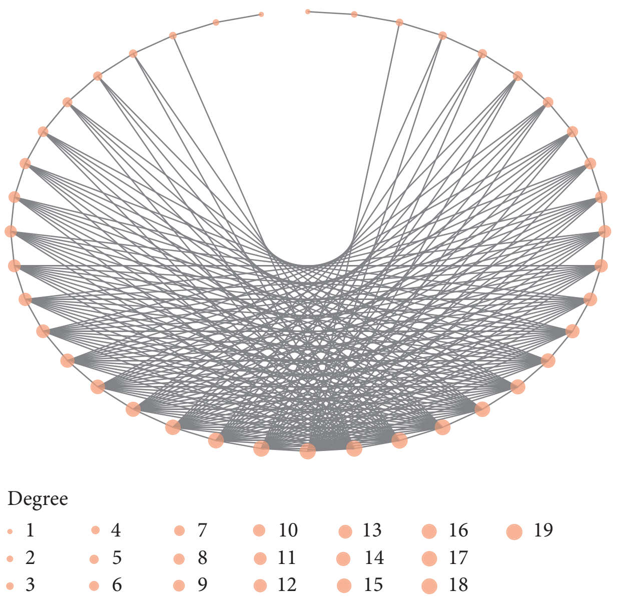


Figure 3: Degree sequence network.

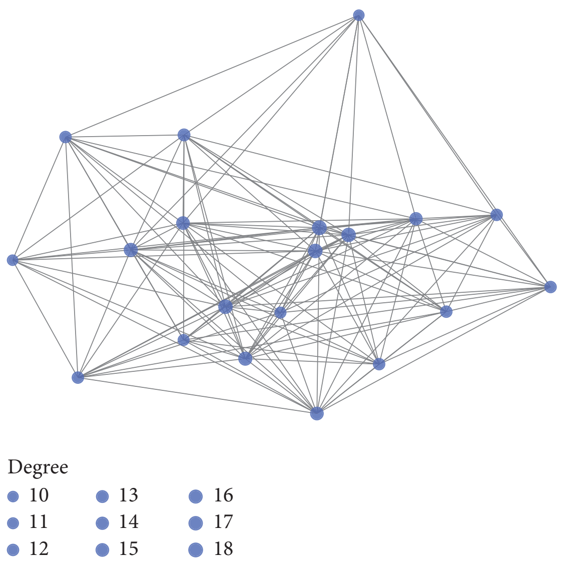


Figure 4: Copeland-Erdös network.

The resulting network is presented in Figure 3. It is very regular, with a uniform distribution of vertex degrees, due to its straightforward generation procedure. However, if one would examine the entropy of the degree sequence, this entropy would be maximal for a given number N of vertices, suggesting far greater randomness of such network. This example shows that entropy of the degree sequence (and the entropy of the degree distribution) can be very misleading when trying to evaluate the true complexity of a network.

2.3.2. Copeland-Erdös Network. The Copeland-Erdös network is a network which seems to be completely random, despite the fact that the procedure of its generation is deterministic. The Copeland-Erdös constant is a constant which is produced by concatenating "0" with the sequence of consecutive prime numbers [25]. When prime numbers are expressed in base 10, the Copeland-Erdös constant is a normal number; that is, its infinite sequence of digits is uniformly distributed (the normality of the Copeland-Erdös constant in bases other than 10 is not proven). This fact allows us to devise the following simple generative procedure for a network. Given the number of vertices N, take the first N^2 digits of the Copeland-Erdös constant and represent them as the matrix of the size $N \times N$. Next, binarize each value in the matrix using the function $f(x) = x \operatorname{div} 5$ (integer division) and use it as the adjacency matrix to create a network. Since each digit in the matrix is approximately equally likely, the resulting binary matrix will have approximately the same number of 0's and 1's. An example of the Copeland-Erdös network is presented in Figure 4. The entropy of the adjacency matrix is maximal for a given number of N vertices; furthermore, the network may seem to be random and complex, but its generative procedure, as we can see, is very simple.

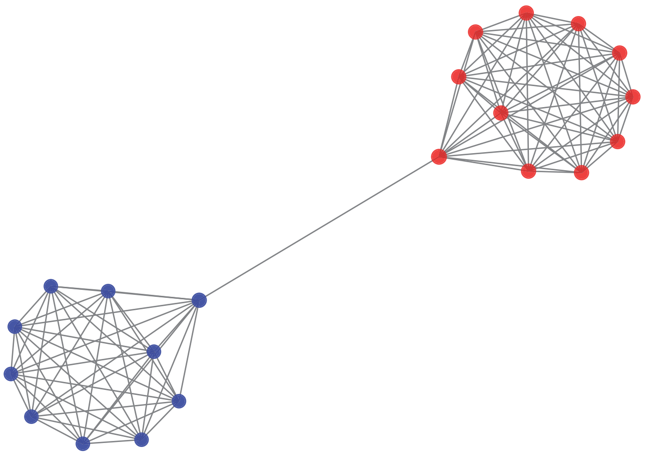


Figure 5: 2-Clique network.

2.3.3. 2-Clique Network. 2-Clique network is an artificial example of a network in which the entropy of the adjacency matrix is maximal. The procedure to generate this network is as follows. We begin with two connected vertices labeled *red* and *blue*. We add *red* and *blue* vertices alternatingly, each time connecting the newly added vertex with all other vertices of the same color. As a result, two cliques appear (see Figure 5). Since there are as many *red* vertices as there are *blue* vertices, the adjacency matrix contains the same number of 0's and 1's (not considering the 1 representing the bridge edge between cliques). So, entropy of the adjacency matrix is close to maximal, although the structure of the network is trivial.

2.3.4. Ouroboros Network. Ouroboros (Ouroboros is an ancient symbol of a serpent eating its own tail, appearing first in Egyptian iconography and then gaining notoriety in later magical traditions) network is another example of an entropy-deceiving network. The procedure to generate this network is very simple: for a given number N of vertices, we create two closed rings, each consisting of $N/2$ vertices, and we connect corresponding vertices of the two rings. Finally, we break a single edge in one ring and we put a single vertex at the end of the broken edge. The result of this procedure can be seen in Figure 6. Interestingly, even though almost all vertices in this network have equal degree of 3, each vertex has different betweenness. Thus, the entropy of the betweenness sequence is maximal, suggesting a very complex pattern of communication pathways though the network. Obviously, this network is very simple from the communication point of view and should not be considered complex.

3. K-Complexity as the Measure of Network Complexity

We strongly believe that Kolmogorov complexity (K-complexity) is a much more reliable and robust basis for constructing the complexity measure for compound objects, such as networks. Although inherently incomputable, K-complexity can be easily approximated to a degree which allows for the practical use of K-complexity in real-world applications, for instance, in machine learning [26, 27], computer network management [28], and general computation theory (proving lower bounds of various Turing machines, combinatorics, formal languages, and inductive inference) [29].

Let us now introduce the formal framework for K-complexity and its approximation. Note that entropy is defined for any random variable, whereas K-complexity is defined for strings of characters only. K-complexity $K_T(s)$ of a string s is formally defined as

$$K_T(s) = \min\{|P|, T(P) = s\}, \quad (9)$$

where P is a program which produces the string s when run on a universal Turing machine T and $|P|$ is the length of the program P, that is, the number of bits required to represent P. Unfortunately, K-complexity is incomputable [30], or more precisely, it is upper semicomputable (only the upper bound of the value of K-complexity can be computed for a given string s). One way for approximating the true value of $K_T(s)$ is to use the notion of algorithmic probability introduced by Solomonoff and Levin [31, 32]. Algorithmic probability $p^a(s)$ of a string s is defined as the expected probability that a random program P running on a universal Turing machine T with the binary alphabet produces the string s upon halting:

$$p^a(s) = \sum_{P:T(P)=s} \frac{1}{2^{|P|}}. \quad (10)$$

Of course there are $2^{|P|}$ possible programs of the length $|P|$, and the summation is performed over all possible programs without limiting their length, which makes algorithmic probability $p^a(s)$ a semimeasure which itself is incomputable. Nevertheless, algorithmic probability can be used to calculate K-complexity using the Coding Theorem [31] which states that algorithmic probability approximates K-complexity up to a constant c:

$$\left|-\log_2 p^a(s) - K_T(s)\right| \leq c. \quad (11)$$

The consequence of the Coding Theorem is that it associates the frequency of occurrence of the string s with its complexity. In other words, if a particular string s can be generated by many different programs, it is considered "simple." On the other hand, if a very specific program is required to produce the given string s, this string can be regarded as "complex." The Coding Theorem also implies that K-complexity of a string s can be approximated from its frequency using the formula:

$$K_T(s) \approx -\log_2 p^a(s). \quad (12)$$

This formula has inspired the Algorithmic Nature Lab group (https://www.algorithmicnaturelab.org) to develop the CTM (Coding Theorem Method), a method to approximate K-complexity by counting output frequencies of small Turing machines. Clearly, algorithmic probability of the string s cannot be computed exactly, because the formula for algorithmic probability requires finding all possible programs that produce the string s. Nonetheless, for a limited subset of Turing machines it is possible to count the number of machines that produce the given string s, and this is the trick behind the CTM. In broad terms, the CTM for a string s consists in computing the following function:

$$\begin{aligned} \mathrm{CTM}(s) &= D(n, m, s) \\ &= \frac{|\{T \in \mathcal{T}(n,m) : T(P) = s\}|}{|\{T \in \mathcal{T}(n,m) : T(P) : \text{halts}\}|}, \end{aligned} \quad (13)$$

where $\mathcal{T}(n,m)$ is the space of all universal Turing machines with n states and m symbols. Function $D(n,m,s)$ computes the ratio of all halting machines with n states and m symbols which produce the string s and its value is determined with the help of known values of the famous Busy Beaver function [33]. The Algorithmic Nature Lab group has gathered statistics on almost 5 million short strings (maximum length is 12 characters) produced by Turing machines with alphabets ranging from 2 to 9 symbols, and based on these statistics the CTM can approximate the algorithmic probability of a given string. Detailed description of the CTM can be found in [34]. Since the function $D(n,m,s)$ is an approximation of the true algorithmic probability $p^a(s)$, it can also be used to approximate K-complexity of the string s.

The CTM can be applied only to short strings consisting of 12 characters or less. For larger strings and matrices, the BDM (Block Decomposition Method) should be used. The BDM requires the decomposition of the string s into (possibly overlapping) blocks $\{b_1, b_2, \ldots, b_k\}$. Given a long string s, the BDM computes its algorithmic probability as

$$\mathrm{BDM}(s) = \sum_{i=1}^{k} \mathrm{CTM}(b_i) + \log_2 |b_i|, \quad (14)$$

where $\mathrm{CTM}(b_i)$ is the algorithmic complexity of the block b_i and $|b_i|$ denotes the number of times the block b_i appears in s. Detailed description of the BDM can be found in [35].

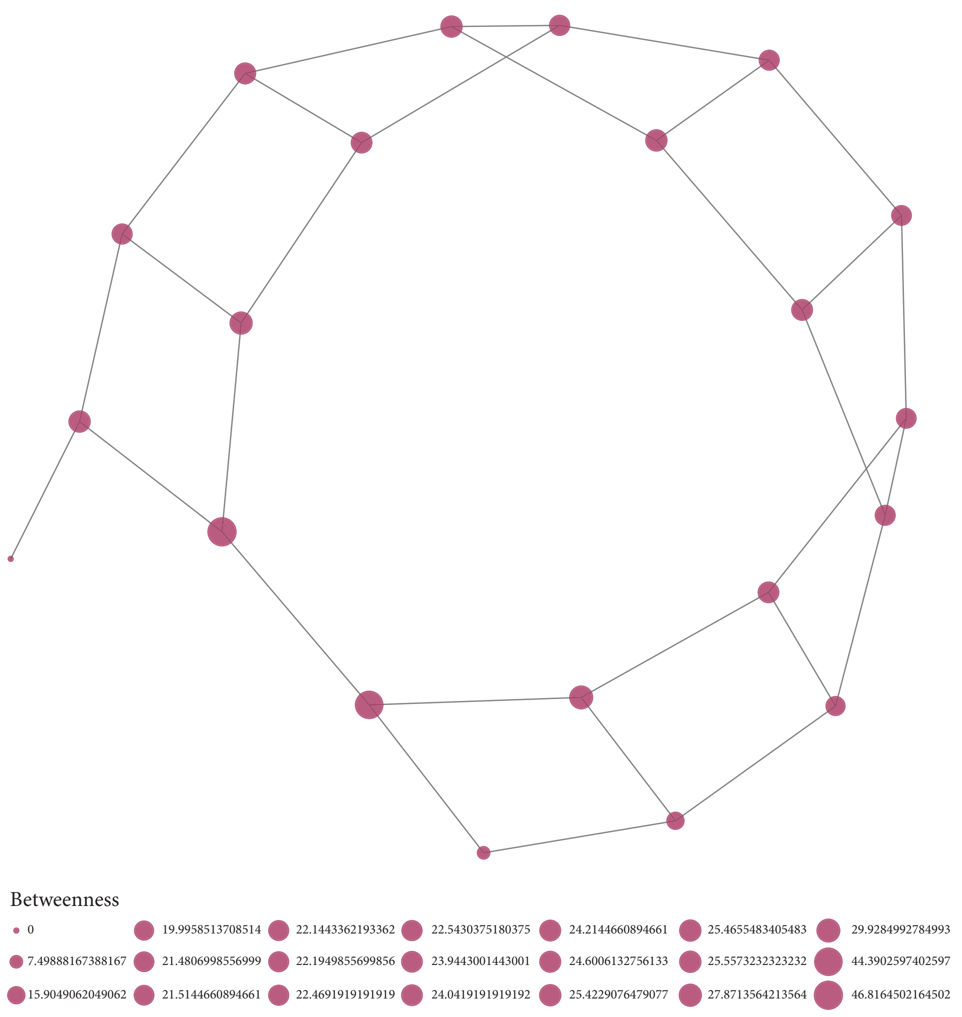


Figure 6: Ouroboros network.

Obviously, any representation of a nontrivial network requires far more than 12 characters. Consider once again the 3-regular graph presented in Figure 1. The Laplacian matrix representation of this graph is the following:

$$L_{10\times 10} = \begin{bmatrix} 3 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 3 & 0 & 0 & -1 & 0 & 0 & -1 & -1 & 0 \\ -1 & 0 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & 0 & -1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & -1 & 0 & 3 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 3 & -1 & 0 & -1 \\ 0 & -1 & 0 & 0 & 0 & -1 & -1 & 3 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & -1 & 3 \end{bmatrix}. \quad (15)$$

If we treat each row of the Laplacian matrix as a separate block, the string representation of the Laplacian matrix becomes $s = \{b_1 = 3010100010, b_2 = 0300100110, \ldots, b_{10} = 0000101013\}$ (for the sake of simplicity, we have replaced the symbol "−1" with the symbol "1"). This input can be fed into the BDM, producing the final estimation of the algorithmic probability (and, consequently, the estimation of the K-complexity) of the string representation of the Laplacian matrix. In our experiments, whenever reporting the values of K-complexity of the string s, we actually report the value of BDM(s) as the approximation of the true K-complexity.

4. Experiments

4.1. Gradual Change of Networks. As we have stated before, the aim of this research is not to propose a new complexity measure for networks, but to compare the usefulness and robustness of entropy versus K-complexity as the underlying foundations for complexity measures. Let us recall what properties are expected from a good and reliable complexity measure for networks. Firstly, the measure should not depend on the particular network representation but should yield more or less consistent results for all possible lossless representations of a network. Secondly, the measure should not equate complexity with randomness. Thirdly, the measure should take into consideration topological properties of a network and not be limited to simple counting of the number of vertices and edges. Of course, statistical properties of a given network will vary significantly between different network invariants, but at the base level of network representation the quantity used to define the complexity measure should fulfill the above requirements. The main question that we are aiming to answer in this study is whether there are qualitative differences between entropy and K-complexity with regard to the above-mentioned requirements when measuring various types of networks.

In order to answer this question we have to measure how a change in the underlying network structure affects the observed values of entropy and K-complexity. To this end, we have devised two scenarios. In the first scenario, the network gradually transforms from the perfectly ordered state to a completely random state. The second transformation brings the network from the perfectly ordered state to a state which can be understood as semiordered, albeit in a different way. The following sections present both scenarios in detail.

4.1.1. From Watts-Strogatz Small-World Model to Erdös-Rényi Random Network Model. A small-world network model introduced by Watts and Strogatz [36] is based on the process, which transforms a fully ordered network with no random edge rewiring into a random network. According to the small-world model, vertices of the network are placed on a regular k-dimensional grid and each vertex is connected to exactly m of its nearest neighbors, producing a regular lattice of vertices with equal degrees. Then, with a small probability p, each edge is randomly rewired. If $p = 0$, no rewiring occurs and the network is fully ordered. All vertices have the same degree, the same betweenness, and the entropy of the adjacency matrix depends only on the density of edges. When $p \geq 0$, edge rewiring is applied to edges and this process distorts the degree distribution of vertices.

On the other end of the network spectrum is the Erdös-Rényi random network model [37], in which there is no inherent pattern of connectivity between vertices. The random network emerges by selecting all possible pairs of vertices and creating, for each pair, an edge with the probability p. Alternatively, one can generate all possible networks consisting of n vertices and m edges and then randomly pick one of these networks. The construction of the random network implies the highest degree of randomness, and there is no other way of describing a particular instance of such network other than by explicitly providing its adjacency matrix or the Laplacian matrix.

In our first experiment, we observe the behavior of entropy and K-complexity being applied to gradually changing networks. We begin with a regular small-world network generated for $p = 0$. Next, we iteratively increase the value of p by 0.01 in each step, until $p = 1$. We retain the network between iterations, so conceptually it is one network undergoing the transition. Also, we only consider rewiring of edges which have not been rewired during preceding iterations, so every edge is rewired at most once. For $p = 0$, the network forms a regular lattice of vertices, and for $p = 1$ the network is fully random with all edges rewired. While randomly rewiring edges, we do not impose any preference on the selection of the target vertex of the edge being currently rewired; that is, each vertex has a uniform probability of being selected as the target vertex of rewiring.

4.1.2. From Watts-Strogatz Small-World Model to Barabási-Albert Preferential Attachment Model. Another popular model of artificial network generation has been introduced by Barabási and Albert [38]. This network model is based on the phenomenon of preferential attachment, according to which vertices appear consecutively in the network and tend to join existing vertices with a strong preference for high degree vertices. The probability of selecting vertex v_i as the target of a newly created edge is proportional to v_i's degree $d(v_i)$. Scale-free networks have many interesting properties [39, 40], but from our point of view the most interesting aspect of scale-free networks is the fact that they represent a particular type of semiorder. The behavior of low-degree vertices is chaotic and random, and individual vertices are difficult to distinguish, but the structure of high-degree vertices (so-called *hubs*) imposes a well-defined topology on the network. High-degree vertices serve as bridges which facilitate communication between remote parts of the network, and their degrees are highly predictable. In other words, although a vast majority of vertices behave randomly, the order appears as soon as high-degree vertices emerge in the network.

In our second experiment, we start from a small-world network and we increment the edge rewiring probability p in each step. This time, however, we do not select the new target vertex randomly, but we use the preferential attachment principle. In the early steps, this process is still random as the differences in vertex degrees are relatively small, but at a certain point the scale-free structure emerges and as more rewiring occurs (for $p \to 1$), the network starts organizing around a subset of high-degree hubs. The intuition is that a good measure of network complexity should be able to distinguish between the initial phase of increasing the randomness of the network and the second phase where the semiorder appears.

4.2. Results and Discussion. We experiment only on artificially generated networks, using three popular network models: Erdös-Rényi random network model, Watts-Strogatz small-world network model, and Barabási-Albert scale-free network model. We have purposefully left out empirical networks from consideration, due to a possible bias which might have been introduced. Unfortunately, for empirical networks, we do not have a good method of approximating the algorithmic probability of a network. All we could do is to compare empirical distributions of network properties (such as degree, betweenness, and local clustering coefficient) with distributions from known generative models. In our previous work [41], we have shown that this approach can lead to severe approximation errors as distributions of network properties strongly depend on values of model parameters (such as edge rewiring probability in the small-world model, or power-law coefficient in the scale-free model). Without a universal method of estimating the algorithmic probability of empirical networks, it is pointless to compare entropy and K-complexity of such networks since no baseline can be established and the results would not yield themselves to interpretation.

In our experiments we have used the `acss` R package [42] which implements the Coding Theorem Method [34, 43] and the Block Decomposition Method [35].

Let us now present the results of the first experiment. In this experiment, the edge rewiring probability p changes from 0 to 1 by 0.01 in each iteration. In each iteration, we generate 50 instances of the network consisting of $N = 100$ vertices, and for each generated network instance, we compute the following measures:

(i) Entropy and K-complexity of the adjacency matrix

(ii) Entropy and K-complexity of the Laplacian matrix

(iii) Entropy and K-complexity of the degree list

(iv) Entropy and K-complexity of the degree distribution

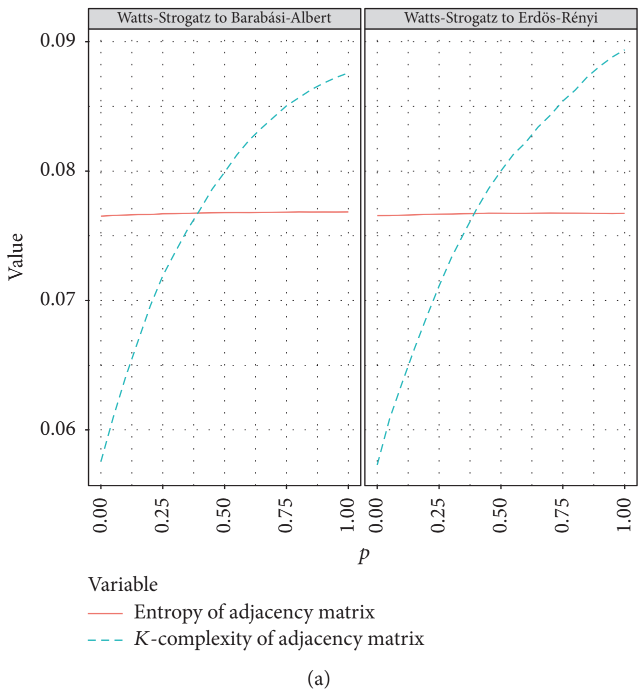
We repeat the experiments described in Section 4.1 for each of the 50 networks, performing the gradual change of each of these networks, and for each value of the edge rewiring probability p we average the results over all 50 networks. Since entropy and K-complexity are expressed in different units, we normalize both measures to allow for side-by-side comparison. The normalization procedure works as follows. For a given string of characters s with the length $l = |s|$, we generate two strings. The first string $s_{\min}$ consists of l repeated 0's and it represents the least complex string of the length l. The second string $s_{\max}$ is a concatenation of l uniformly selected digits and it represents the most complex string of the length l. Each value of entropy and K-complexity is normalized with respect to minimum and maximum value of entropy and K-complexity possible for a string of equal length. This allows us not only to compare entropy and K-complexity between different representations of networks, but also to compare entropy to K-complexity directly. The results of our experiments are presented in Figure 7.

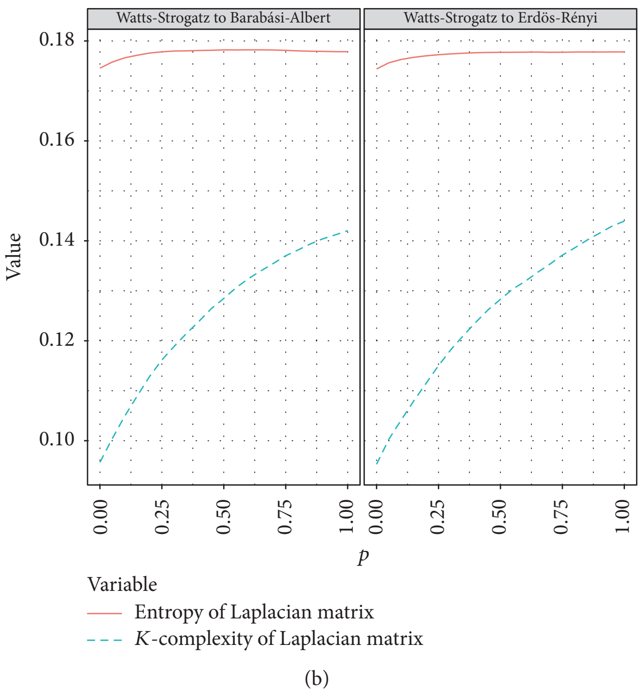
We observe that traditional entropy of the adjacency matrix remains constant. This is obvious, the rewiring of edges does not change the density of the network (the number of edges in the original small-world network and the final random network or scale-free network is exactly the same), so entropy of the adjacency matrix is the same for each value of the edge rewiring probability p. On the other hand, K-complexity of the adjacency matrix slowly increases. It should be noted that the change of K-complexity is small when analyzed in absolute values. Nevertheless, K-complexity consistently increases as networks diverge from the order of the small-world model toward the chaos of random network model. A very similar result can be observed for networks represented using Laplacian matrices. Again, entropy fails to signal any change in network's complexity because the density of networks remains constant throughout the transition, and the very slight change of entropy for $p \in \langle 0, 0.25 \rangle$ is caused by the change of the degree list which forms the main diagonal of the Laplacian matrix. The result for the degree list is more surprising. K-complexity of the degree list slightly increases as networks lose their ordering but remains close to 0.4. At the same time, entropy increases quickly as the edge rewiring probability p approaches 1. The pattern of entropy growth is very similar for both the transition to random network and the transition to scale-free network, with the latter characterized counterintuitively by larger entropy. In addition, the absolute value of entropy for the degree list is several times larger than for the remaining network representations (the adjacency matrix and the Laplacian matrix). Finally, both entropy and K-complexity behave similarly for networks described using degree distributions. We note that both measures correctly identify the decrease of apparent complexity as networks approach the scale-free model (when semiorder emerges) and signal increasing complexity as networks become more and more random. It is tempting to conclude from the results of the last experiment that the degree distribution is the best representation when network complexity is concerned. However, one should not forget that the degree distribution and the degree list are not lossless representations of networks, so the algorithmic complexity of degree distribution only estimates how difficult it is to recreate that distribution and not the entire network.

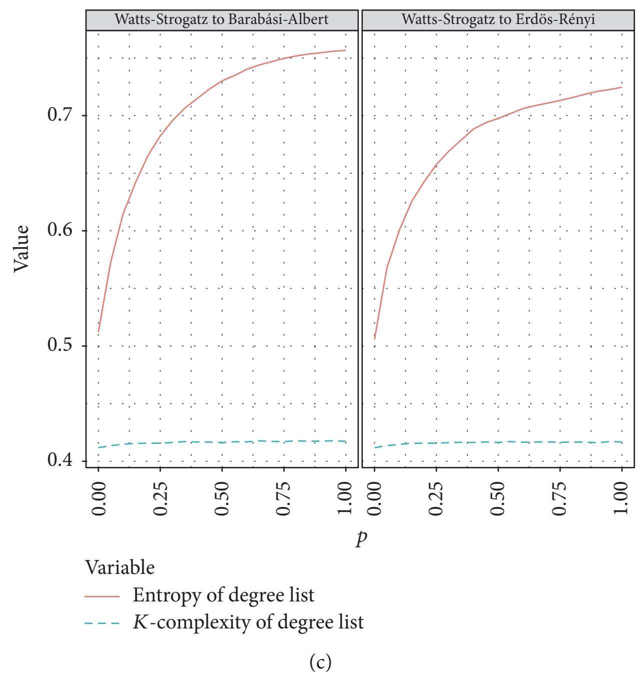
Given the requirements formulated at the beginning of this section and the results of the experimental evaluation, we conclude that K-complexity is a more feasible measure for constructing intuitive complexity definitions. K-complexity captures small topological changes in the evolving networks, where entropy cannot detect these changes due to the fact that network density remains constant. Also, K-complexity produces less variance in absolute values across different network representations, and entropy returns drastically different estimates depending on the particular network representation.

5. Conclusions

Entropy has been commonly used as the basis for modeling the complexity of networks. In this paper, we show why entropy may be a wrong choice for measuring network complexity. Entropy equates complexity with randomness and requires preselecting the network feature of interest. As we have shown, it is relatively easy to construct a simple network which maximizes entropy of the adjacency matrix, the degree sequence, or the betweenness distribution. On the other hand, K-complexity equates the complexity with the length of the computational description of the network. This measure is much harder to deceive and it provides a more robust and reliable description of the network. When networks gradually transform from the highly ordered to highly disordered states, K-complexity captures this transition, at least with respect to adjacency matrices and Laplacian matrices.







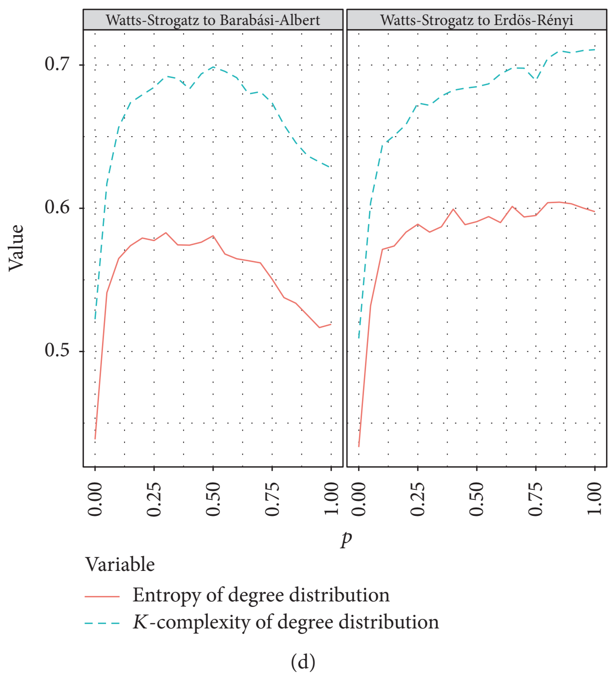


FIGURE 7: Entropy and K-complexity of (a) adjacency matrix, (b) Laplacian matrix, (c) degree list, and (d) degree distribution under gradual transition from Watts-Strogatz model to Erdös-Rényi and Barabási-Albert models.

In this paper, we have used traditional methods to describe a network: the adjacency matrix, the Laplacian matrix, the degree list, and the degree distribution. We have limited the scope of experiments to three most popular generative network models: random networks, small-world networks, and scale-free networks. However, it is possible to describe networks more succinctly, using universal network generators. In the near future, we plan to present a new method of computing algorithmic complexity of networks without having to estimate K-complexity, but rather following the minimum description length principle. Also, extending the experiments to the realm of empirical networks could prove to be informative and interesting. We also intend to investigate network representations based on various energies (Randić energy, Laplacian energy, and adjacency matrix energy) and to research the relationships between network energy and K-complexity.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank Adrian Szymczak for helping to devise the degree sequence network. This work was supported by the National Science Centre, Poland, Decisions nos. DEC-2016/23/B/ST6/03962, DEC-2016/21/B/ST6/01463, and DEC-2016/21/D/ST6/02948; European Union's Horizon 2020 Research and Innovation Program under the Marie Skłodowska-Curie Grant Agreement no. 691152 (RENOIR); and the Polish Ministry of Science and Higher Education Fund for supporting internationally cofinanced projects in 2016–2019 (Agreement no. 3628/H2020/2016/2).

References

[1] L. Todd Veldhuizen, "Software libraries and their reuse: Entropy, kolmogorov complexity, and zipf's law," *Library-Centric Software Design (LCSD'05)*, p. 11, 2005.

[2] D. Bonchev and G. A. Buck, "Quantitative measures of network complexity," in *Complexity in chemistry, biology, and ecology*, Math. Comput. Chem., pp. 191–235, Springer, New York, 2005.

[3] J. Cardoso, J. Mendling, G. Neumann, and H. A. Reijers, "A discourse on complexity of process models," in *Business Process Management Workshops*, vol. 4103 of *Lecture Notes in Computer Science*, pp. 117–128, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[4] J. Cardoso, "Complexity analysis of BPEL Web processes," *Software Process Improvement and Practice*, vol. 12, no. 1, pp. 35–49, 2007.

[5] A. Latva-Koivisto, 2001, Finding a complexity measure for business process models.

[6] G. M. Constantine, "Graph complexity and the Laplacian matrix in blocked experiments," *Linear and Multilinear Algebra*, vol. 28, no. 1-2, pp. 49–56, 1990.

[7] D. L. Neel and M. E. Orrison, "The linear complexity of a graph," *Electronic Journal of Combinatorics*, vol. 13, no. 1, Research Paper 9, 19 pages, 2006.

[8] J. Kim and T. Wilhelm, "What is a complex graph?" *Physica A. Statistical Mechanics and its Applications*, vol. 387, no. 11, pp. 2637–2652, 2008.

[9] M. Dehmer, F. Emmert-Streib, Z. Chen, X. Li, and Y. Shi John Wiley & Sons, Mathematical foundations and applications of graph entropy, 2016.

[10] M. Dehmer and A. Mowshowitz, "A history of graph entropy measures," *Information Sciences. An International Journal*, vol. 181, no. 1, pp. 57–78, 2011.

[11] A. Mowshowitz and M. Dehmer, "Entropy and the complexity of graphs revisited," *Entropy. An International and Interdisciplinary Journal of Entropy and Information Studies*, vol. 14, no. 3, pp. 559–570, 2012.

[12] S. Cao, M. Dehmer, and Y. Shi, "Extremality of degree-based graph entropies," *Information Sciences. An International Journal*, vol. 278, pp. 22–33, 2014.

[13] Z. Chen, M. Dehmer, and Y. Shi, "A note on distance-based graph entropies," *Entropy. An International and Interdisciplinary Journal of Entropy and Information Studies*, vol. 16, no. 10, pp. 5416–5427, 2014.

[14] K. C. Das and S. Sorgun, "On randic energy of graphs," *MATCH Commun. Math. Comput. Chem*, vol. 72, no. 1, pp. 227–238, 2014.

[15] I. Gutman and B. Zhou, "Laplacian energy of a graph," *Linear Algebra and its Applications*, vol. 414, no. 1, pp. 29–37, 2006.

[16] C. E. Shannon, *The Mathematical Theory of Communication*, The University of Illinois Press, Urbana, Ill, USA, 1949.

[17] A. N. Kolmogorov, "Three approaches to the quantitative definition of information," *International Journal of Computer Mathematics. Section A. Programming Theory and Methods. Section B. Computational Methods*, vol. 2, pp. 157–168, 1968.

[18] A. Rényi et al., "On measures of entropy and information," in *Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 547–561, University of California Press, 1961.

[19] A. Mowshowitz, "Entropy and the complexity of graphs: I. An index of the relative complexity of a graph," *The Bulletin of Mathematical Biophysics*, vol. 30, no. 1, pp. 175–204, 1968.

[20] I. Ji, W. Bing-Hong, W. Wen-Xu, and Z. Tao, "Network entropy based on topology configuration and its computation to random networks," *Chinese Physics Letters*, vol. 25, no. 11, p. 4177, 2008.

[21] M. Dehmer, "Information processing in complex networks: graph entropy and information functionals," *Applied Mathematics and Computation*, vol. 201, no. 1-2, pp. 82–94, 2008.

[22] S. Cao, M. Dehmer, and Z. Kang, "Network entropies based on independent sets and matchings," *Applied Mathematics and Computation*, vol. 307, pp. 265–270, 2017.

[23] J. Körner, "Fredman-komlós bounds and information theory," *SIAM Journal on Algebraic Discrete Methods*, vol. 7, no. 4, pp. 560–570, 1986.

[24] H. Zenil, N. A. Kiani, and J. Tegnér, "Low-algorithmic-complexity entropy-deceiving graphs," *Physical Review E*, vol. 96, no. 1, 2017.

[25] N. Sloane, "The on-line encyclopedia of integer sequences," https://oeis.org/A033308.

[26] C. Faloutsos and V. Megalooikonomou, "On data mining, compression, and Kolmogorov complexity," *Data Mining and Knowledge Discovery*, vol. 15, no. 1, pp. 3–20, 2007.

[27] J. Schmidhuber, "Discovering neural nets with low Kolmogorov complexity and high generalization capability," *Neural Networks*, vol. 10, no. 5, pp. 857–873, 1997.

[28] A. Kulkarni and S. Bush, "Detecting distributed denial-of-service attacks using Kolmogorov Complexity metrics," *Journal of Network and Systems Management*, vol. 14, no. 1, pp. 69–80, 2006.

[29] M. Li and P. M. B. Vitányii, "Kolmogorov complexity and its applications," in *Algorithms and Complexity*, Texts and Monographs in Computer Science, pp. 1–187, Springer-Verlag, New York, 2014.

[30] G. J. Chaitin, "On the length of programs for computing finite binary sequences," *Journal of the Association for Computing Machinery*, vol. 13, pp. 547–569, 1966.

[31] L. A. Levin, "Laws on the conservation (zero increase) of information, and questions on the foundations of probability theory," *Akademiya Nauk SSSR. Institut Problem Peredachi Informatsii Akademii Nauk SSSR. Problemy Peredachi Informatsii*, vol. 10, no. 3, pp. 30–35, 1974.

[32] R. J. Solomonoff, "A formal theory of inductive inference. Part I," *Information and Control*, vol. 7, no. 1, pp. 1–22, 1964.

[33] T. Rado, "On non-computable functions," *The Bell System Technical Journal*, vol. 41, pp. 877–884, 1962.

[34] F. Soler-Toscano, H. Zenil, J.-P. Delahaye, and N. Gauvrit, "Calculating Kolmogorov complexity from the output frequency distributions of small turing machines," *PLoS ONE*, vol. 9, no. 5, Article ID e96223, 2014.

[35] H. Zenil, F. Soler-Toscano, N. A. Kiani, S. Hernández-Orozco, and A. Rueda-Toicen, *A decomposition method for global evaluation of shannon entropy and local estimations of algorithmic complexity*, 2016.

[36] D. J. Watts and S. H. Strogatz, "Collective dynamics of "small-world" networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[37] P. Erdös and A. Rényi, "On the evolution of random graphs," *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, pp. 17–61, 1960.

[38] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *American Association for the Advancement of Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[39] A.-L. Barabási, "Scale-Free Networks: a Decade and beyond," *American Association for the Advancement of Science. Science*, vol. 325, no. 5939, pp. 412-413, 2009.

[40] M. E. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[41] T. Kajdanowicz and M. Morzy, "Using Kolmogorov Complexity with Graph and Vertex Entropy to Measure Similarity of Empirical Graphs with Theoretical Graph Models," in *Proceedings of the 2nd International Electronic Conference on Entropy and Its Applications*, p. C003, Sciforum.net.

[42] N. Gauvrit, H. Singmann, F. Soler-Toscano, and H. Zenil, "Algorithmic complexity for psychology: a user-friendly implementation of the coding theorem method," *Behavior Research Methods*, vol. 48, no. 1, pp. 314–329, 2016.

[43] J.-P. Delahaye and H. Zenil, "Numerical evaluation of algorithmic complexity for short strings: A glance into the innermost structure of randomness," *Applied Mathematics and Computation*, vol. 219, no. 1, pp. 63–77, 2012.