mathematics

MDPI

Article

Random Networks with Quantum Boolean Functions

Mario Franco [1,2], Octavio Zapata [1], David A. Rosenblueth [1,3] and Carlos Gershenson [1,3,4,*]

1 Centro de Ciencias de la Complejidad, Universidad Nacional Autónoma de México, Ciudad de Mexico 04510, Mexico; mfrancom88@ciencias.unam.mx (M.F.); octavio.zapata@c3.unam.mx (O.Z.); drosenbl@unam.mx (D.A.R.)
2 Posgrado en Ciencia e Ingeniería de la Computación, Universidad Nacional Autónoma de México, Ciudad de Mexico 04510, Mexico
3 Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, Universidad Nacional Autónoma de México, Ciudad de Mexico 04510, Mexico
4 Lakeside Labs GmbH, Lakeside Park B04, 9020 Klagenfurt am Wörthersee, Austria
* Correspondence: cgg@unam.mx

Simple Summary: Using the paradigm of quantum computation, we propose a novel model similar to random Boolean networks. We study the properties and dynamics of our model and find that these differ non-trivially from those of traditional random Boolean networks.

Abstract: We propose quantum Boolean networks, which can be classified as deterministic reversible asynchronous Boolean networks. This model is based on the previously developed concept of quantum Boolean functions. A quantum Boolean network is a Boolean network where the functions associated with the nodes are quantum Boolean functions. We study some properties of this novel model and, using a quantum simulator, we study how the dynamics change in function of connectivity of the network and the set of operators we allow. For some configurations, this model resembles the behavior of reversible Boolean networks, while for other configurations a more complex dynamic can emerge. For example, cycles larger than 2^N were observed. Additionally, using a scheme akin to one used previously with random Boolean networks, we computed the average entropy and complexity of the networks. As opposed to classic random Boolean networks, where "complex" dynamics are restricted mainly to a connectivity close to a phase transition, quantum Boolean networks can exhibit stable, complex, and unstable dynamics independently of their connectivity.

Keywords: random Boolean networks; reversible Boolean networks; quantum computing

Citation: Franco, M.; Zapata, O.; Rosenblueth, D.A.; Gershenson, C. Random Networks with Quantum Boolean Functions. *Mathematics* **2021**, *9*, 792. https://doi.org/10.3390/math9080792

Academic Editor: J. Alberto Conejero

Received: 1 February 2021
Accepted: 3 March 2021
Published: 7 April 2021

1. Introduction

Boolean networks were originally proposed in the late 1960s to model the behavior of regulatory networks by Kauffman [1]. Boolean networks have been used since to model the operation of discrete dynamical systems. This model is suitable for the study of complex systems as it offers a simple and flexible way to control the interaction of the elements of a system and observe the effects of structure change in the network dynamics. At the same time, quantum algorithms are still not well understood and it is as yet unclear for which kind of problem quantum computers will provide us with some sort of advantage. Quantum computation may prove beneficial to enrich the model of Boolean networks, considering that the dynamics described by quantum mechanics is fundamentally different from that of classical dynamics. Our objective is to explore the possibility of developing, within the quantum-computing framework, a model similar to that of classical Boolean networks.

In the 1980s, when the idea of using one of the smallest parts of nature to compute was proposed by Benioff [2], the field of quantum computing was born. One of the principles of quantum theory is that the evolution in time of a system which is isolated from its environment is given by a first-order differential equation called the Schrödinger equation.

By solving this equation, one obtains a time-evolution operator which is necessarily unitary (see e.g., [3], Section 6.4.1). By definition, unitary operators are reversible. This means that it is not possible to have two different states evolve into the exact same state—as with dissipative systems such as classic Boolean networks—since it would no longer be possible to determine the initial state from the final state. In other words, information cannot be destroyed.

Nonetheless, quantum computing was not the cradle of reversible computing. Reversible computing formally appeared in 1961 with Landauer's work [4]. In his research, Landauer linked the irreversibility of conventional computational operations and the thermodynamic behavior of the computational device. He stated that whenever a bit is overwritten with a new value, the previous information is not physically destroyed. Even if this information were lost for all practical purposes, it still exists as heat that was pushed out of the computational device. Landauer's reasoning can be acknowledged if one considers that some of the most fundamental laws of physics are reversible.

Reversible computing has received limited attention, since it is quite complicated to implement and in many cases it is possible to get along without it. However, it has been slowly gaining followers [5], mainly due to the recently grim predictions casted upon conventional computing: in the near future, making a transistor smaller will no longer yield any practical improvement [6]. In addition, it is impossible to reduce the size of a transistor forever. Moreover, since reversibility is ubiquitous at the quantum level, reversible models may be useful to model some discrete physical systems.

When we take into consideration all the aforementioned, the idea of a quantum flavor of the Boolean network model naturally sprouts.

This paper is structured as follows: Section 2 serves to recall some useful concepts related to Boolean networks and provides a brief introduction to quantum computing. Section 2.4 provides a definition of a quantum Boolean function and a method for generating random quantum Boolean functions. In Section 3.1, we define the model of quantum Boolean networks and state a few basic properties. In Section 3, we generate quantum Boolean networks at random and study the average cycle length, the entropy and the complexity of the networks. Finally, Section 4 concludes the paper.

2. Preliminaries

The columns of a square, $n \times n$ matrix A are denoted by $\boldsymbol{a}_j$ for $j = 1, 2 \ldots, n$. The $n \times n$ matrix A^T is called the *transpose* of A and is defined as follows:

$$A^T = \begin{bmatrix} \boldsymbol{a}_1^T \\ \vdots \\ \boldsymbol{a}_n^T \end{bmatrix}$$

If the entries of A are complex numbers, then A^* denotes the conjugate transpose $\bar{A}^T$. We say that A is *invertible* if there is a matrix A^{-1} such that $A^{-1}A = I$, where I is the identity matrix. A complex square matrix U is called *unitary* if $U^*U = I$. If U is a $n \times n$ unitary matrix, then its columns $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$ are pairwise orthogonal vectors of norm 1: for all $i, j = 1, \ldots, n$, we have $\boldsymbol{u}_i^* \boldsymbol{u}_j = 0$ if $i \neq j$ and $\|\boldsymbol{u}_j\| = \sqrt{\boldsymbol{u}_j^* \boldsymbol{u}_j} = \boldsymbol{u}_j^* \boldsymbol{u}_j = 1$ if $i = j$.

Let us consider a system with two degrees of freedom. This corresponds to choosing the standard basis vectors

$$\boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

for some two-dimensional Euclidean space. A linear combination $\boldsymbol{p} = x\boldsymbol{e}_1 + y\boldsymbol{e}_2$ with $x, y \in \mathbb{R}$ satisfying $x, y \geq 0$ (non-negativity) and $x + y = 1$ (affiness) is called *convex*. The components x and y of the convex combination

$$\boldsymbol{p} \;=\; x\begin{bmatrix}1\\0\end{bmatrix} + y\begin{bmatrix}0\\1\end{bmatrix} \;=\; \begin{bmatrix}x\\y\end{bmatrix}$$

represent the probabilities of observing the system at states $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$, respectively. Hence, any convex combination of the basis elements is a discrete probability distribution on the set of states of the system.

2.1. Quantum Computation

Here we use standard, linear algebra notation. This is, however, not the usual practice in quantum theory literature. For a complete introduction to the subject, we refer the reader to [7].

A quantum system with two degrees of freedom is called a *qubit*. A *state* of a qubit is a unit vector ψ of the complex vector space $\mathbb{C}^2$. In this context, the components of ψ are called *amplitudes* and represent the square-roots of the probabilities of observing the system at each of the states of the standard basis of $\mathbb{C}^2$. If α is a complex number of norm 1, the unit vectors $\alpha\psi$ and ψ determine the same state up to a *phase factor*.

Although single qubits are useful for illustrating some of the elemental quantum phenomena, in practice, they are of little interest. We can combine multiple qubits to create a multi-qubit system. This is done via the tensor product ($\otimes$) of the basis vectors corresponding to the individual vector spaces. For example, the state of a two qubit system will be given by:

$$\psi = \alpha\boldsymbol{e}_1 \otimes \boldsymbol{e}_1 + \beta\boldsymbol{e}_1 \otimes \boldsymbol{e}_2 + \delta\boldsymbol{e}_2 \otimes \boldsymbol{e}_1 + \gamma\boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \tag{1}$$

with $|\alpha|^2 + |\beta|^2 + |\delta|^2 + |\gamma|^2 = 1$. The reader may notice that this combination automatically gives us a basis for the state space of the new system, i.e., the vectors $\boldsymbol{e}_1 \otimes \boldsymbol{e}_1, \boldsymbol{e}_1 \otimes \boldsymbol{e}_2, \boldsymbol{e}_2 \otimes \boldsymbol{e}_1$, and $\boldsymbol{e}_2 \otimes \boldsymbol{e}_2$ form an orthonormal basis of $\mathbb{C}^4$. In general, any linear combination of basis states where the sum of the modulus square of the coefficients is 1 is called a *superposition*.

Another important concept in quantum computation is entanglement. We say that two systems are *entangled* if they are not separable or, in other words, we say that a system is entangled if it cannot be decomposed as a product of its subsystems. For example, it is not complicated to show that for all $\alpha, \beta, \gamma, \delta \in \mathbb{C}$:

$$\frac{1}{\sqrt{2}}\boldsymbol{e}_1 \otimes \boldsymbol{e}_1 + \frac{1}{\sqrt{2}}\boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \neq (\alpha\boldsymbol{e}_1 + \beta\boldsymbol{e}_2) \otimes (\delta\boldsymbol{e}_1 + \gamma\boldsymbol{e}_2)$$

The evolution in discrete time of a quantum system that do not interact with its environment is implemented by multiplying a unitary matrix with the state vector. Given a unitary matrix U and a complex vector ψ of norm $\|\psi\| = 1$, we have $\|U\psi\|^2 = (U\psi)^* U\psi = \psi^*(U^*U)\psi = \psi^*\psi = 1$. Therefore, $\|U\psi\| = 1$ so U preserves the norm of unit vectors. Hence, U maps quantum states to quantum states. Unitary evolution is reversible because the inverse of a unitary matrix U always exists and it is equal to its conjugate transpose, i.e., $U^{-1} = U^*$. The matrix

$$A \;=\; \frac{1}{2}\begin{bmatrix}0 & 1 & 0 & 1\\ 1 & 0 & 1 & 0\\ 0 & 1 & 0 & 1\\ 1 & 0 & 1 & 0\end{bmatrix}$$

has non-negative entries and the sum of each of its rows is equal to 1, i.e., $A\boldsymbol{e} = \boldsymbol{e}$ where $\boldsymbol{e}$ is the all-ones vector. Non-negative matrices satisfying this condition are called *stochastic*. Since A is symmetric, i.e., $A^T = A$, we have $(A\boldsymbol{e})^T = \boldsymbol{e}^T A = \boldsymbol{e}^T$. That is, all the columns of A also sum to 1. Thus, A is *doubly stochastic*. Hence, A maps discrete distributions to

discrete distributions. However, A is not unitary because it has columns that are neither orthogonal nor unit vectors: $\boldsymbol{a}_1 = \boldsymbol{a}_3$, $\boldsymbol{a}_2 = \boldsymbol{a}_4$, and $\|\boldsymbol{a}_j\| = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$ for $j = 1, \ldots, 4$.

Therefore, the probabilistic dynamics given by a doubly stochastic matrix is in general not unitary.

2.2. Random Boolean Networks

A *random Boolean network* (RBN) is a tuple $(\Sigma, \mathcal{F})$, where $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ is a set of Boolean variables taking values in $\{0, 1\}$, and $\mathcal{F}$ is a set of n Boolean functions $f_i \colon \{0,1\}^{k_i} \to \{0,1\}$ with $k_i \leq n$ for $i = 1, \ldots, n$; each f_i is chosen uniformly at random from all $\sum_{k \leq n} 2^{2^k}$ possible Boolean functions of arity at most n. This model can be represented as a directed graph $G = (V, E)$, see Figure 1, where $V = \{1, \ldots, n\}$ and there is a directed edge $(j, i) \in E$ if and only if f_i is a function of the Boolean variable σ_j associated with the node j. RBNs have been extensively studied in the last decades and multiple generalizations have been proposed [8–12].

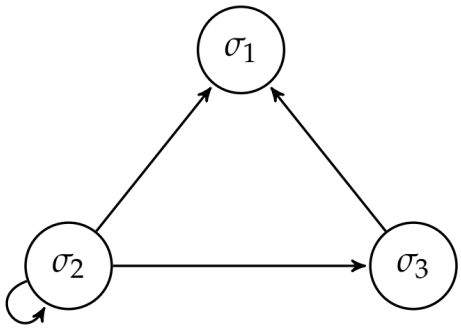
We can simulate a deterministic dynamical process using an RBN as follows. At the start, each σ_i is assigned a value at random and at each $t = 0, 1, \ldots$ its value is updated according to the evaluation of f_i which depends on some subset of Σ. Concretely, suppose $\boldsymbol{x} \in \{0,1\}^n$ is an n-dimensional vector containing random values drawn uniformly from the set $\{0, 1\}$. Each component of $\boldsymbol{x}$ is the initial state of a Boolean variable in the network:

$$\boldsymbol{x}_i = \sigma_i^{\{0\}}$$

for $i = 1, \ldots, n$. For each timestep t, the transition of the ith variable σ_i is defined by the following updating scheme:

$$\sigma_i^{\{t\}} \quad \mapsto \quad \sigma_i^{\{t+1\}} = f_i(\sigma_{i_1}^{\{t\}}, \ldots, \sigma_{i_{k_i}}^{\{t\}})$$

where $\{i_1, \ldots, i_{k_i}\}$ is the subset of nodes of V such that $(i_r, i) \in E$ for all $r = 1, \ldots, k_i$.



$$\sigma_1^{\{t+1\}} = f_1(\sigma_2^{\{t\}}, \sigma_3^{\{t\}}) \qquad \sigma_2^{\{t+1\}} = f_2(\sigma_2^{\{t\}}) \qquad \sigma_3^{\{t+1\}} = f_3(\sigma_2^{\{t\}})$$

Figure 1. Visualization of a Boolean network as a graph.

Another important aspect of Boolean networks is its update scheme. Many Boolean networks have a *synchronous* update scheme, i.e., all nodes are updated at the same time but, in general, we can also have Boolean networks that update only a fraction of its nodes at each timestep, *asynchronous*. This update can be made in a particular order each time, *deterministic*, or randomly, *non-deterministic*. Let us notice that synchronous networks are a special case of asynchronous networks (when the subset of nodes updated each timestep is Σ) and deterministic networks are a special case of non-deterministic networks (when the update scheme always occurs in the same order).

2.3. Reversible Networks

Reversible Boolean networks (RevBN) [13,14] are of particular interest for the present work, as will become clear next. A *reversible Boolean network* is a tuple $(\Omega, \Sigma, \mathcal{F})$, where $\Omega = \{\omega_1, \ldots, \omega_n\}$ and $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ are sets of Boolean variables, $\sigma_i, \omega_i \in \{0,1\}$, $i = 1, \ldots, n$, and $\mathcal{F}$ is a set of n Boolean functions $f_i \colon \{0,1\}^{k_i} \to \{0,1\}$ with $k_i \leq n$, chosen uniformly from all the possible Boolean functions of at most arity n. Again, we define a dynamical process on these random networks. At the start, each variable σ_i, ω_i is assigned a value at random, and a two-dimensional state vector $\boldsymbol{s}_i = \sigma_i \boldsymbol{e}_1 + \omega_i \boldsymbol{e}_2$ is prepared for all i:

$$\boldsymbol{s}_i = \begin{bmatrix} \sigma_i \\ \omega_i \end{bmatrix}$$

The state of the ith node of the network is the first component of the state vector $\boldsymbol{s}_i$. Therefore, the state of the whole network is not a 1-dimensional vector but a $2 \times n$ matrix with columns $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n$. Each timestep $t = 0, 1, \ldots$ the network variables are updated, in a synchronous way, changing each state vector, as follows:

$$\boldsymbol{s}_i^{\{t\}} = \begin{bmatrix} \sigma_i^{\{t\}} \\ \omega_i^{\{t\}} \end{bmatrix} \quad \mapsto \quad \boldsymbol{s}_i^{\{t+1\}} = \begin{bmatrix} \omega_i^{\{t\}} \oplus f_i(\sigma_{i_1}^{\{t\}}, \ldots, \sigma_{i_{k_i}}^{\{t\}}) \\ \sigma_i^{\{t\}} \end{bmatrix}$$

where $\oplus$ represents addition modulo 2. This description extends the state space of the network, and doing so, it avoids the degenerated cycles defined in the original papers about reversible Boolean networks.

In this kind of networks all states are part of an attractor, i.e., there are no basins of attraction. Moreover, since every state of the network is part of an attractor, it can be shown that this implies that the network is reversible since every state of an attractor is mapped to exactly one state at the next timestep.

Another way to visualize this reversibility is that, independently of the computation the network has performed, it is possible to recover the initial state. At any time step $t+1$, we can easily recover $\sigma_i^{\{t\}}$ from Ω, since $\omega_i^{\{t+1\}} = \sigma_i^{\{t\}}$. Recovering $\omega_i^{\{t\}}$ is also straightforward since $\omega_i^{\{t\}} = \sigma_i^{\{t+1\}} \oplus f_i(\sigma_{i_1}^{\{t\}}, \ldots, \sigma_{i_{k_i}}^{\{t\}})$. Notice that we can compute $f_i(\sigma_{i_1}^{\{t\}}, \ldots, \sigma_{i_{k_i}}^{\{t\}})$ without any trouble because, from the previous step, we already know all $\sigma_i^{\{t\}}$. Furthermore, considering the previous development and the fact that the network has a finite amount of possible states it follows that the state space of the network is a finite collection of disjoint cycles.

When reversible Boolean networks were originally proposed, the authors mentioned that in this type of network there are different types of attractors. We believe this is misleading, they disregard the effect of what we like to call the "memory of the network". As a consequence, we observe a somewhat richer dynamic in attractors, for example, the network starts to traverse the cycle backwards.

2.4. Quantum Boolean Functions

In order to define an analogous model of Boolean networks within the quantum realm, we use the notion of quantum Boolean function from [15]. A *quantum Boolean function* on k qubits is a $2^k \times 2^k$ unitary matrix U such that $UU = I$. It follows from this definition, that any quantum Boolean function U satisfies $U^* = U$. Additionally, if U and V are quantum Boolean functions on the same number of qubits, then UV is a quantum Boolean function if and only if $UV = VU$. A standard way to encode a k-ary Boolean function

$f\colon \{0,1\}^k \longrightarrow \{0,1\}$ as a quantum Boolean function, is to define the following diagonal matrix:

$$U_f = \begin{bmatrix} (-1)^{f(0,0,\ldots,0)} & & & \\ & (-1)^{f(0,0,\ldots,1)} & & \\ & & \ddots & \\ & & & (-1)^{f(1,1,\ldots,1)} \end{bmatrix}$$

Since U_f is a diagonal matrix with ± 1 in all diagonal entries, then $U_f U_f$ is a diagonal matrix with $(\pm 1)^2 = 1$ in all diagonal entries, i.e., $U_f U_f = I$. The two 2×2 unitary matrices

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \text{and} \qquad H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

are called the Pauli X and Hadamard operators. These two operators are quantum Boolean functions on one qubit: $XX = I$ and $HH = I$. Their action on the standard basis vectors of $\mathbb{C}^2$ can be written as follows:

$$\begin{aligned} \boldsymbol{e}_1 &\mapsto X\boldsymbol{e}_1 = \boldsymbol{e}_2 & \qquad \boldsymbol{e}_1 &\mapsto H\boldsymbol{e}_1 = {}^1/\sqrt{2}(\boldsymbol{e}_1 + \boldsymbol{e}_2) \\ \boldsymbol{e}_2 &\mapsto X\boldsymbol{e}_2 = \boldsymbol{e}_1 & \qquad \boldsymbol{e}_2 &\mapsto H\boldsymbol{e}_2 = {}^1/\sqrt{2}(\boldsymbol{e}_1 - \boldsymbol{e}_2) \end{aligned}$$

Although the definition of quantum Boolean function is rather general, it may not be useful for us without some simple restrictions. The first thing we may notice is that this formulation of quantum Boolean functions can map a state to any other state. Such mapping can be troublesome in the context of Boolean networks since we want functions to change only one node, not the whole network at the same time. This problem can be easily solved if we restrict ourselves to the subset of quantum Boolean functions that can only affect one qubit at a time. As an example, let us consider the following mapping:

$$\begin{aligned} \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 &\mapsto \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 & \qquad \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 &\mapsto \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \\ \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 &\mapsto \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 & \qquad \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 &\mapsto \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 &\mapsto \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 & \qquad \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 &\mapsto \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \\ \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 &\mapsto \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 & \qquad \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 &\mapsto \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 \otimes \boldsymbol{e}_1 \end{aligned}$$

This mapping is implemented by the following block diagonal 8×8 matrix:

$$B = \begin{bmatrix} X & & & \\ & X & & \\ & & I & \\ & & & X \end{bmatrix}$$

This kind of mapping only affects the last qubit (in a reversible way) by preserving or inverting the qubit conditioned on the other input qubits. It is not difficult to see that B is quantum Boolean. This way of declaring quantum Boolean functions resembles the disjunctive normal form of Boolean functions in which the function is defined via some conjunction of terms with the difference that, in this case, we apply the Pauli X operator to the qubit if the input is a term of the function.

We can replace the X operator for any quantum Boolean operator U that acts on a single qubit and the complete operator will still be a quantum Booelan operator. For example, we can replace the X operator with the Hadamard operator. This operator puts the qubit into a balanced superposition of $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. Considering the previous mapping,

if we replace some of the implicit X operators for the H operator, we obtain a quantum Boolean function that can create superpositions of states:

$$\begin{aligned}
e_1 \otimes e_1 \otimes e_1 &\mapsto e_1 \otimes e_1 \otimes e_2 & e_1 \otimes e_2 \otimes e_1 &\mapsto 1/\sqrt{2}(e_1 \otimes e_2 \otimes e_1 + e_1 \otimes e_2 \otimes e_2) \\
e_1 \otimes e_1 \otimes e_2 &\mapsto e_1 \otimes e_1 \otimes e_1 & e_1 \otimes e_2 \otimes e_2 &\mapsto 1/\sqrt{2}(e_1 \otimes e_2 \otimes e_1 - e_1 \otimes e_2 \otimes e_2) \\
e_2 \otimes e_1 \otimes e_1 &\mapsto e_2 \otimes e_1 \otimes e_1 & e_2 \otimes e_2 \otimes e_1 &\mapsto 1/\sqrt{2}(e_2 \otimes e_2 \otimes e_1 + e_2 \otimes e_2 \otimes e_2) \\
e_2 \otimes e_1 \otimes e_2 &\mapsto e_2 \otimes e_1 \otimes e_2 & e_2 \otimes e_2 \otimes e_2 &\mapsto 1/\sqrt{2}(e_2 \otimes e_2 \otimes e_1 - e_2 \otimes e_2 \otimes e_2)
\end{aligned} \tag{2}$$

Even more, we can set aside the target qubit to write a representation of this kind of mappings that is similar to the truth table representation of Boolean functions. For example, we can represent the previous mapping with the following table:

In general, let $U_1, \ldots, U_{2^{n-1}}$ be quantum Boolean function on a single qubit and let

$$B = \begin{bmatrix} U_1 & & & \\ & U_2 & & \\ & & \ddots & \\ & & & U_{2^{n-1}} \end{bmatrix}$$

Since B is diagonal and each U_i is quantum Boolean, then B is a quantum Boolean function on n qubits. The previous development only allows the creation of quantum Boolean operators that modify the last qubit using the input in a particular order. We can easily compute any Boolean operator that targets any qubit using any input order by simply rearranging the qubits order via a change of basis. Such particular change of basis is performed with a permutation matrix. Recall that a *permutation matrix* is a square matrix with a unique 1 entry per row and per column, and 0 everywhere else. Now, let U be a quantum Boolean function and let $B = P^T U P$ for some P permutation matrix. By definition, permutation matrices are orthogonal, i.e., $P^T P = I$. Hence, $BB = (P^T U P)(P^T U P) = P^T U (P P^T) U P = P^T (UU) P = I$. Therefore, B is a quantum Boolean function.

3. Random Quantum Boolean Networks

3.1. Quantum Boolean Networks

A *quantum Boolean network* (QBN) is a tuple $(\Sigma, \mathcal{F}, \Pi)$, where Σ is a set of n qubits, $\mathcal{F} = \{U_1, \ldots, U_n\}$ is a set of quantum Boolean operators and $\Pi = \{\pi_1, \ldots, \pi_n\}$ is a permutation of the set of indices $\{1, \ldots, n\}$ that serves as the update order of the network. Each U_i is a quantum Boolean function on $1 \leq k_i \leq n$ qubits that only modifies the state of the ith qubit. The last element is required to specify the behavior of the network without any ambiguity since, in general, the Boolean operators do not commute. Specifically, the transition matrix B of the network is the product:

$$B = U_{\pi_1} \cdots U_{\pi_n}$$

Since the product of quantum Boolean functions is quantum Boolean, B is unitary. Thus, quantum Boolean networks are reversible and have an update scheme that is asynchronous and deterministic. Just to clarify, a complete timestep of the network is equivalent to the product of all the operators in the specified order:

$$\psi^{\{t+1\}} = U_{\pi_1} \cdots U_{\pi_n} \psi^{\{t\}} = B \psi^{\{t\}} \tag{3}$$

The state of the network is a linear combination of all the vectors spanned by the tensor product of the qubits. In other words, it may be any vector of norm one in $\mathbb{C}^{2^n}$. In QBNs every state is part of an attractor because the evolution is reversible, i.e., every state comes from a particular state and goes to another one; otherwise the computation will not

be reversible. Another curious property of QBNs is that they can generate cycles of infinite length with a finite amount of nodes. Consider the following operator:

$$U = \begin{bmatrix} cos(\theta) & sin(\theta) \\ sin(\theta) & -cos(\theta) \end{bmatrix} \tag{4}$$

Again, it is a simple exercise to show that U is a quantum Boolean function of arity 1. Moreover, U is a rotation parameterized by θ. In theory, we can set θ equal to some irrational number which in turn will produce a rotation by some irrational angle and consequently, no matter how many times we apply U again, we will never return to the initial configuration.

One important distinction between QBNs with any other classical Boolean network is that every node is always connected to itself. This follows from the fact that, in quantum computing, we cannot erase information and the previous state of the node always affects the computation of next state of itself. Therefore, the minimum connectivity of the every node in the graph is one.

Owing to the fact that we are interested in studying the general behavior of this new kind of Boolean network, we adopt a similar scheme to the one of random Boolean networks, i.e., randomly generate Boolean networks along the space of possible networks and study how a change in certain parameters changes the average dynamic of the networks.

To generate random networks with n nodes, we require a set of n quantum Boolean operators and a permutation of the first n integers which will serve as an order. We generate the former by first computing a relationship graph $G = (V, E)$ between the nodes in such a way that

$$P((i,j) \in E \mid i,j \in V) = \frac{K}{N-1}$$

where K is the desired connectivity of the network and N is the number of nodes. Although the previous way of computing the graph does not always produces a graph with connectivity exactly equal to K, in practice, it produces graphs with average connectivity quite close to the target value. Afterwards, we create a set of N quantum Boolean operators with arity equal to the number of edges pointing to the target node. Each operator is computed by associating each input label with a single qubit quantum Boolean operator (see Table 1 for an example) and then use such a mapping to define a unitary complex matrix (similarly to the construction done in the last paragraph of Section 2.4). For the latter, we simply select a permutation of the set $\{1, \ldots, n\}$ uniformly.

Table 1. Tabular representation of the quantum mapping of (2).

σ_1	σ_2	U
0	0	X
0	1	H
1	0	I
1	1	H

Although we previously restricted the set of all possible quantum Boolean functions to the subset of functions that only modifies the state of at most one qubit, this set is still not useful from a practical point of view (most of these functions are generic rotations without a direct computational meaning which may create cycles of infinite length). We further refine this set to the one of quantum Boolean functions that can be constructed with the procedure we stated before using only the Pauli operators (I, X, Y, Z) plus Hadamard (H). In practice, it is complicated to implement an arbitrary operator and the Pauli operators are among the easiest operators to implement. In addition, they can also be interpreted more easily, for example, the X operator is the generalization of the bit flip operation and

the H operator can be seen as a split of the bit into its two possible values. These operators are implemented by the following two 2×2 matrices:

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

While this set may seem rather small, it is quite powerful, it is capable of phenomena like phase change, superposition, and entanglement can emerge. Given these five operators, we can naturally design four experimental sets: $\{I, X\}$, $\{I, X, Y, Z\}$, $\{I, X, H\}$, $\{I, X, Y, Z, H\}$. Intuitively, the first set can only flips bits conditioned on the input; the second, can flip and alter the phase of the system; the third and fourth, are variations of the previous configurations but this time we allow the creation of superposition (the third set can induce a real phase to the state but not a complex phase). Additionally, due to the fact that simulations of large quantum registers are computationally prohibitive, we restricted our simulation to small networks. For the simulation, we used the quantum simulator QuEST [16].

3.2. Cycle Lengths

In Figure 2, we show some examples of the dynamics present in QBNs. The figure shows the magnitude of amplitude of all states of the computation basis at timestep t. Although this way of illustrating the dynamics is completely blind to quantum phase we consider it quite illustrative. When we prohibit the creation of superpositions we observe a dynamic that is quite similar to the dynamic of RevBNs. On the other hand, when we allow the creation of such states, we observe a much more complex dynamic emerge.

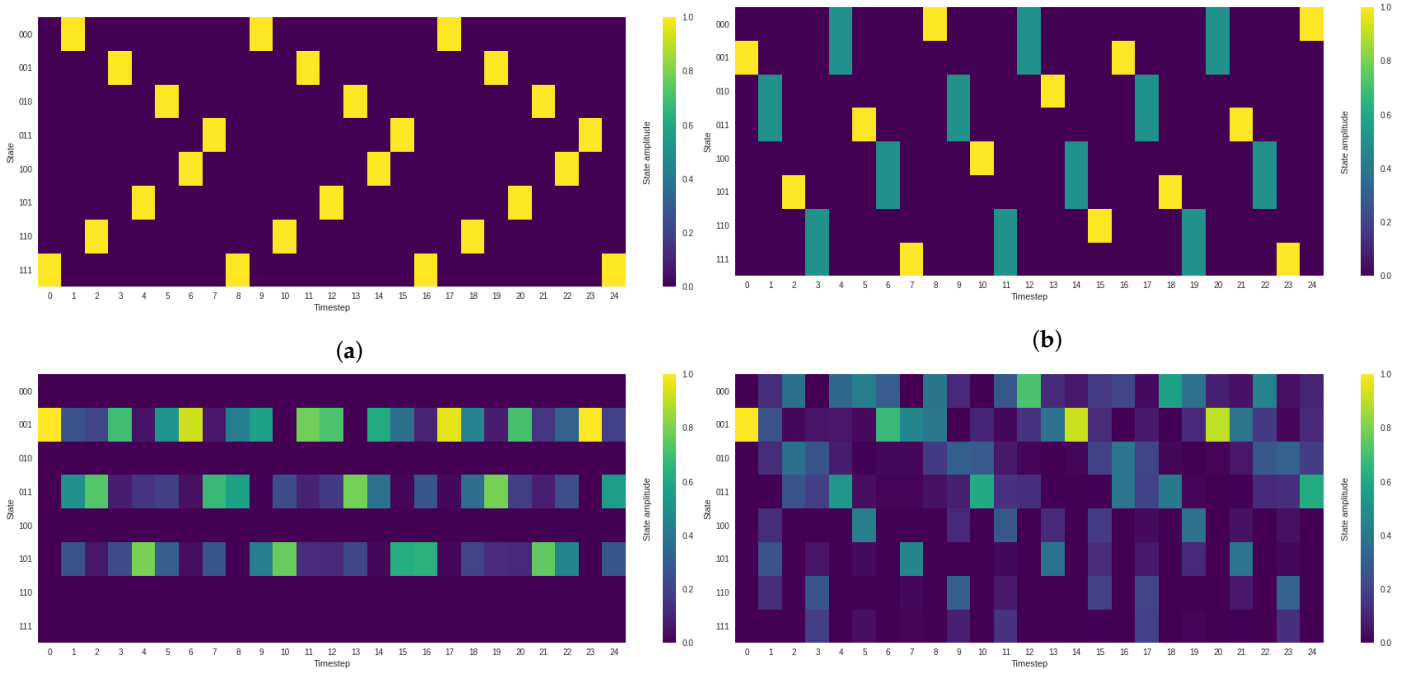


Figure 2. Examples of different dynamics exhibited by a quantum Boolean network (QBN) with 3 nodes. In panel (**a**), we observe a dynamic that keeps the amplitude concentrated in one particular state, such dynamic resemblances the behavior of a reversible Boolean network. Panels (**b**–**d**) illustrate some of the dynamics that can be observed in QBNs.

When we compute the survival probability of cycles in the network (the probability that a cycle does not return to the initial state at the next timestep $t + 1$ given that its current size is l), see Figure 3, it becomes clear that we can now produce cycles larger than 2^N. Just by including operators that induce a phase, Y, and Z, to the state, it is possible to obtain

larger cycles. Furthermore, an explosion in the length of the cycles can be observed when we introduce the possibility to create superposition of states, via the the H operator.

The first phenomenon, the possibility of larger cycles when we include operators that can induce a phase on the quantum state, emerges simply as a mathematical curiosity and is quite straightforward to explain. For example, take the operators X and Y, both behave similarly:

$$X\boldsymbol{e}_1 = \boldsymbol{e}_2 \qquad Y\boldsymbol{e}_1 = i\boldsymbol{e}_2 \tag{5}$$

but when we compare both states we have that they are strictly different. Similarly, if we take the operator that describes the complete dynamic of the network, B, it may happen that, somewhere along the cycle, B induces a phase on the state and then, we return to the initial state, we have two states that are strictly different. Considering the operator sets we are using, it is possible to create cycles four times larger because this sets allow four possible phases: $+1$, -1, $+i$, and $-i$. Experimentally we do not observe cycles that are large, probably because they require a specific amount of phase flips and our building method may be biased away from those configurations.

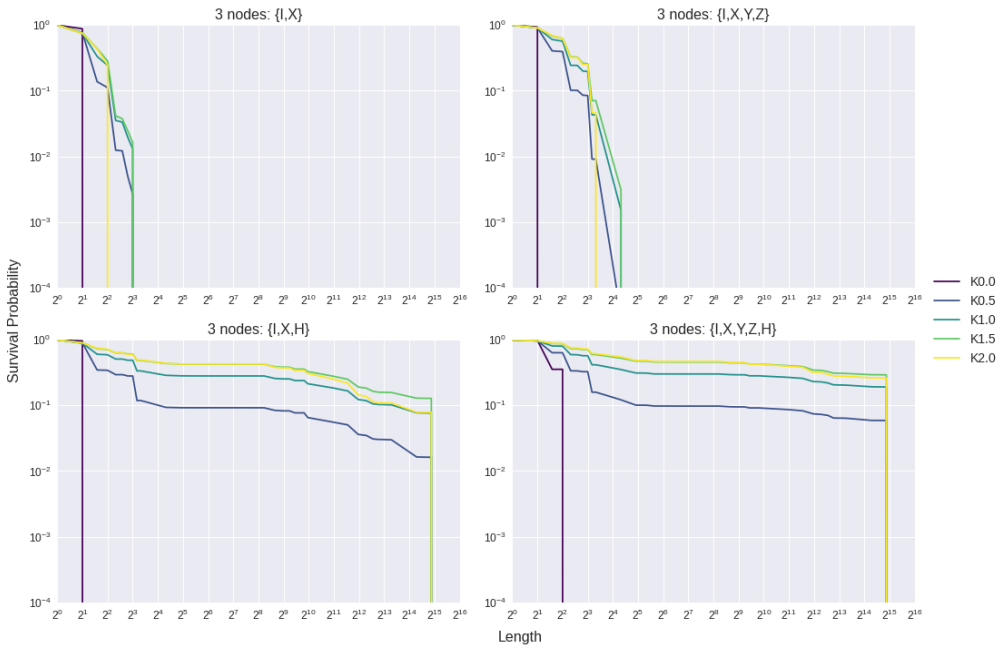
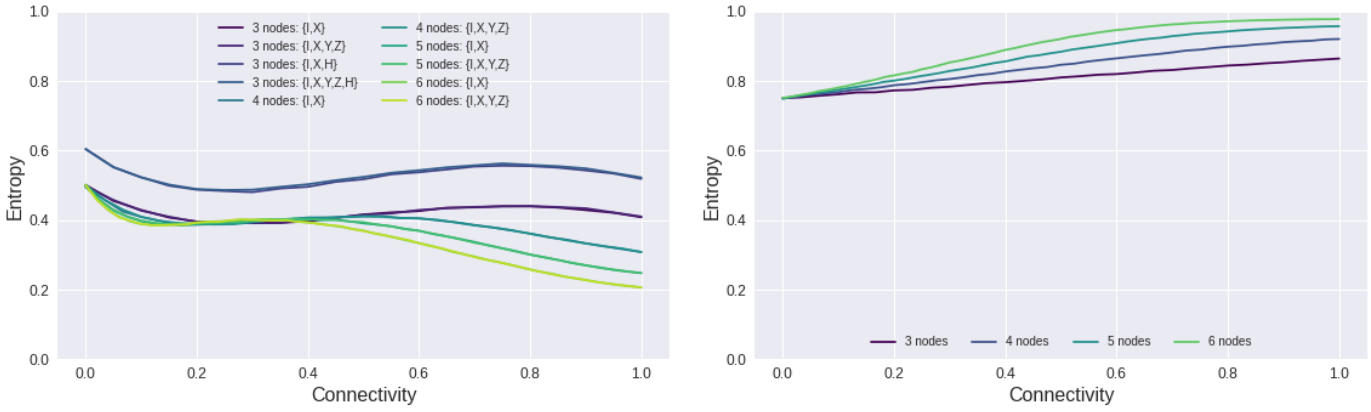


Figure 3. Survival probability for different connectivity values of a network with 3 nodes. The survival probability is computed from 20,000 realizations. The survival probability for reversible Boolean networks (RevBNs) and QBNs for different network sizes can be consulted in Figures A1 and A2.

The cycle length explosion is also easily explained. A quantum state is a linear combination of states, any linear combination of magnitude one, and Hadamard allow us to grasp a subset of such infinite set of linear combinations. Unfortunately, we were only able to study the effect of superposition for a really small case ($N = 3$) since a mixture between the enormous increase in length and the propagation of errors make it impossible to compute cycle lengths with confidence.

3.3. Entropy

We also computed the Shannon entropy (S) for different operator sets and network sizes, see Figure 4. The entropy is computed as the average entropy of multiple QBNs realizations. The entropy of a particular QBN is calculated as the average entropy of its cycles and the entropy of a cycle is just the average Shannon entropy of the nodes, using the node measurement probability of being $\boldsymbol{e}_1$ or $\boldsymbol{e}_2$.



(**a**) (**b**)

Figure 4. Entropy for different QBN configurations is shown in panel (**a**). In order to allow a comparison between networks of different sizes, we normalized the connectivity value. Panel (**b**) shows the entropy for RevBNs of different sizes. A detailed version of the entropy for QBNs and RevBNs can be consulted in Figures A3 and A4, respectively.

It is relatively simple to calculate the entropy when the connectivity is equal to zero. Here, we first analyze the case when the set is $\{I, X\}$ and all operators are selected with uniform probability. Let us denote the probability that the network is composed exactly of k Pauli X operators as $P(X = k)$ then,

$$P(X = k) = \frac{N!}{k!(N-k)!}(p_X)^k(p_I)^{N-k} \tag{6}$$

With this operator set and connectivity value, we can only invert the state of a node and this change is reverted the next timestep. It is easy to calculate the entropy of the network given the number of X's gates.

$$S(network \mid X = k) = \frac{k}{N} \tag{7}$$

Finally, to obtain the average entropy of the network, we just compute the weighted mean,

$$\mathbb{E}(S) = \sum_{k=1}^{N} P(X = k)S(network \mid X = k) = \sum_{k=1}^{N} \frac{(N-1)!}{(k-1)!(N-k)!}(p_X)^k(p_I)^{N-k} \tag{8}$$

A little test for the case when $N = 3$ and the operators are selected with uniform probability,

$$\mathbb{E}(S) = \sum_{k=1}^{3} \frac{2!}{(k-1)!(3-k)!}\left(\frac{1}{2}\right)^3 = 4\left(\frac{1}{2}\right)^3 = \frac{1}{2} \tag{9}$$

In general, for the case when the connectivity is equal to zero, we have that the probability of a particular combination of operators is,

$$P(U_1 = k_1 \wedge \cdots \wedge U_m = k_m) = \frac{N!}{k_1! \cdots k_m!}\prod_{i=1}^{m}(p_{U_i})^{k_i} \tag{10}$$

Now, since the entropy is given by the average entropy of the nodes, we can compute the entropy individually according to the operator that is applied to the qubit,

$$S(U_1 = k_1 \wedge \cdots \wedge U_m = k_m) = \frac{1}{N}\sum_{i=1}^{m} S(U_i = k_i) \tag{11}$$

Therefore, the average entropy is,

$$\mathbb{E}(S) = \sum_{k_1=1}^{m} \cdots \sum_{k_m=m-(k_{m-1}+\cdots+k_1)}^{m} \left(\frac{N!}{k_1! \cdots k_m!} \prod_{i=1}^{m} (p_{U_i})^{k_i} \right) \left(\frac{1}{N} \sum_{i=1}^{m} S(U_i = k_i) \right) \quad (12)$$

If we consider a network with three nodes constructed with the set $\{I, X, H\}$, the previous formula gives us an estimate for the entropy of 0.603, which fits exactly with the mean of the observations. Considering the complex behavior that emerges when we increase the connectivity value, it is rather difficult to say anything more from a theoretical perspective.

3.4. Complexity

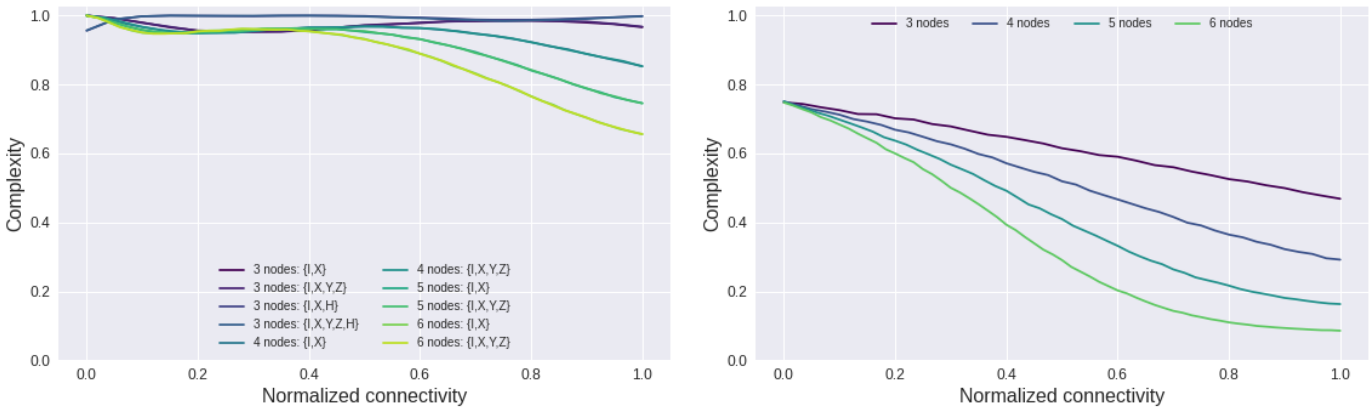
In addition to cycle lengths and entropies, we also computed the complexity of the networks [17], see Figure 5. The complexity of a network is given by:

$$C(S) = 4S(1 - S) \quad (13)$$

with S the entropy of the network. The factor 4 is used to normalize the measure to the interval $[0, 1]$. Intuitively, complexity aims to measure the balance between change and stability. It is minimal for $S = 0$ (no change, only one state possible) and also for $S = 1$ (constant change, all states equally possible). Complexity is maximal $C = 1$ when $S = 0.5$, which coincides with phase transitions in classical random Boolean networks and the Ising model, as well as with class IV cellular automata [18].

We observed that QBNs have a higher complexity than RevBNs. This may be interpreted as QBNs being less extreme in their dynamics, in other words, they present a behavior that is more balanced in terms of change and regularity than RevBNs. In both cases, the complexity tends to decrease at higher connectivity values suggesting that the dynamic in both cases is moving towards a chaotic or an ordered regime. Additionally, bigger networks have less complexity at greater connectivities. We suspect this is because smaller networks have a smaller state space and there is not much of a difference between regular and irregular in a small space.

It is worth noticing that QBNs have a high complexity average almost independently of the connectivity. This contrasts with classical RBNs, where a high complexity is found only near the phase transition ($K = 2$). In addition, the high standard deviations (see Figure A5) suggest that the precise dynamic behavior of a QBN depends more on the operators it uses and less on the connectivity. RevBNs would be an intermediate case, where complexity and standard deviations are higher than those of classical RBNs, but lower than those of QBNs (see Figure A6).



(**a**) (**b**)

Figure 5. Panel (**a**) shows the complexity for different QBN configurations. Connectivity was normalized to allow a direct comparison between networks of different sizes. Panel (**b**) show the complexity for RevBNs of different sizes. A detailed version of the complexity for QBNs and RevBNs can be consulted in Figures A5 and A6, respectively.

4. Conclusions

In the present work, we proposed a new Boolean network model using the paradigm of quantum computation: quantum Boolean networks, which exhibit a deterministic, asynchronous, and reversible dynamic. QBNs can create networks with an odd number of nodes in contrast with RevBNs which require an auxiliary set of nodes of the same size in order to allow reversible evolution; thus, RevBNs always have an even number of nodes. If we consider a non-degenerated version of the state space that a reversible Boolean network spans (a state space that also includes the memory nodes), for a restricted set of operators, the unitary dynamics of our model behaves similarly to the dynamics of reversible Boolean networks. We conjecture that QBNs are a generalization of RevBNs but a formal proof eluded us for the time being.

This model is likely to be better suited to model phenomena in which we cannot neglect quantum effects. Additionally, we suspect it may be possible to encode our lack of certainty about the current state of the network as a superposition, then let the network evolve and recover the most probable states at a certain time but further research about which operators better represent classical dynamics and inclusion of dissipation may be required, perhaps as measurements or a Hamiltonian which slowly drives the dynamics to low energy states. Independently of this, including any form of information dissipation may result in an appealing version of QBN, especially because it can bring closer this model and the dissipative variants of Boolean networks.

One interesting point about this model is that it also permits to control the "quantum-ness" of the network via restrictions to the set of single qubit quantum Boolean functions. However, this does not provide a fine-grained level of control.

The way we say that two states are equal could be debatable. Although, formally they are two different things, in practice it is quite complicated to distinguish between two states that only differ up to a global phase (a phase that can be factorized from every state of the basis). Therefore, it is possible to argue that we cannot produce larger cycles than 2^N by means of phase alone. Nonetheless, we consider this possibility mathematically interesting. In addition, when we have a superposition of states, we can induce a relative phase between states with the advantage that such phase can be distinguished and also used to create larger cycles.

Author Contributions: Conceptualization, M.F., O.Z., C.G., D.A.R.; methodology, M.F., O.Z., C.G., D.A.R.; software, M.F.; validation, M.F.; formal analysis, M.F., O.Z., C.G., D.A.R.; investigation, M.F.; writing—original draft preparation, M.F.; writing—review and editing, M.F., O.Z., C.G., D.A.R.; visualization, M.F.; supervision, C.G., O.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

RBN	random Boolean network
RevBN	reversible Boolean network
QBN	quantum Boolean network

Appendix A

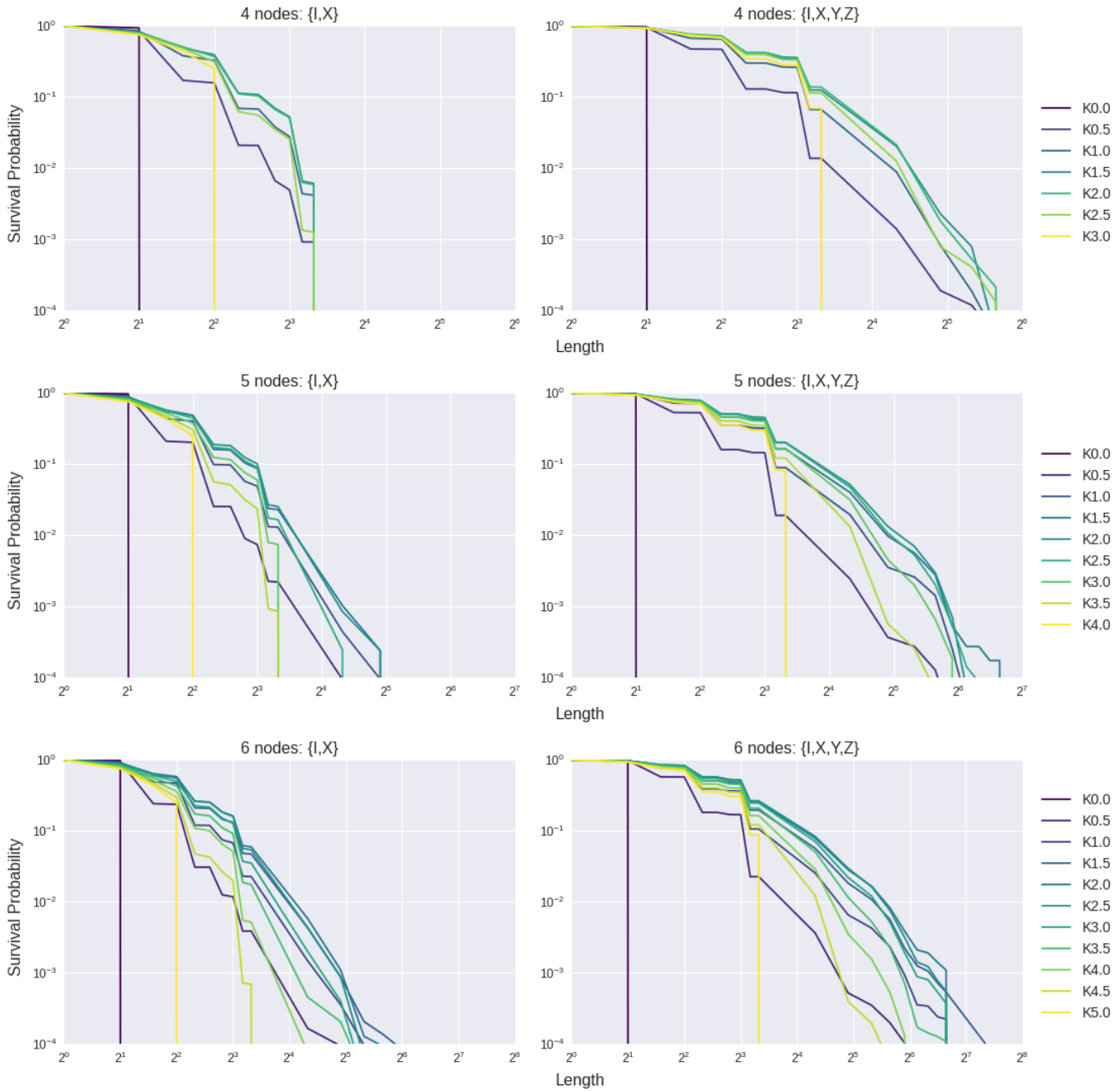


Figure A1. Survival probability for different connectivity values of QBNs with 3, 4, 5, and 6 nodes. The survival probability was computed from 20,000 realizations. For practical reasons (extremely large cycles and computational errors), configurations that can produce superpositions were omitted.

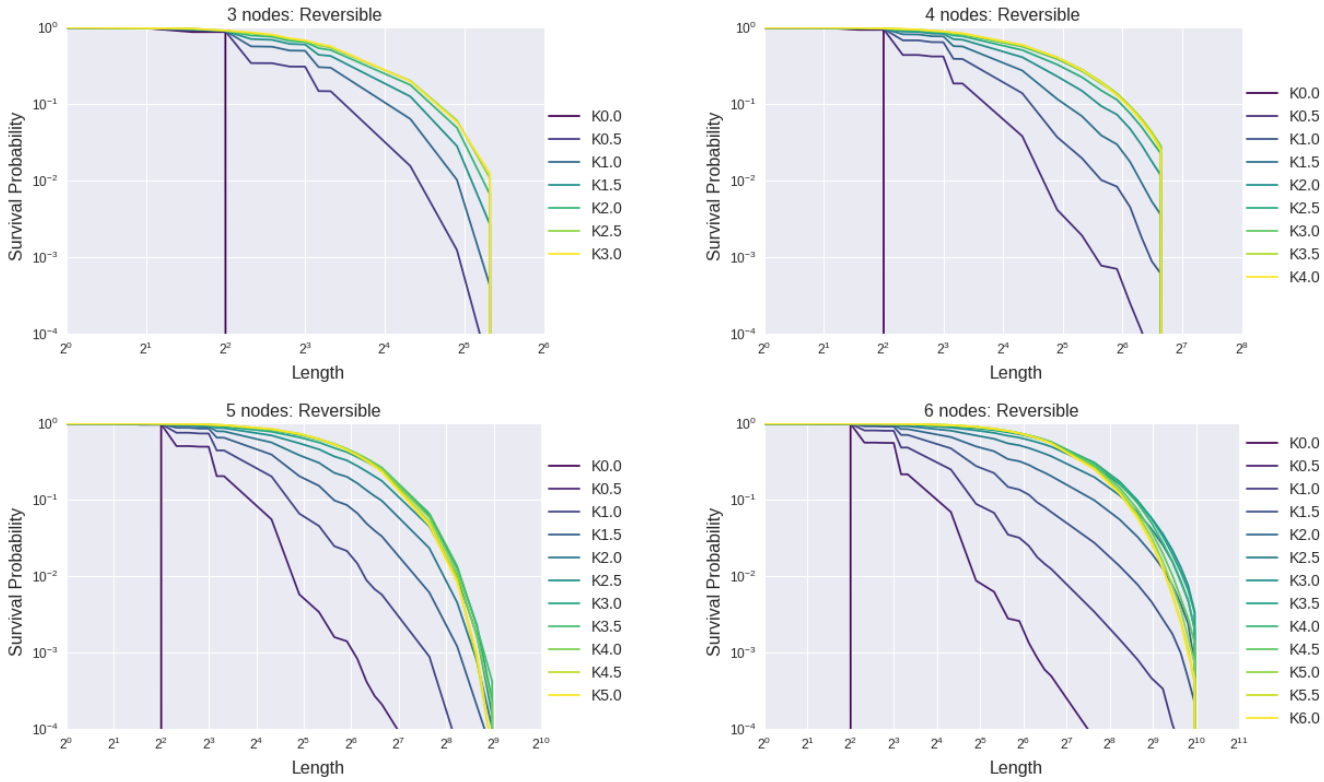


Figure A2. Survival probability for different connectivity values of RevBNs with 3, 4, 5, and 6 nodes. The survival probability was computed from 20,000 realizations.

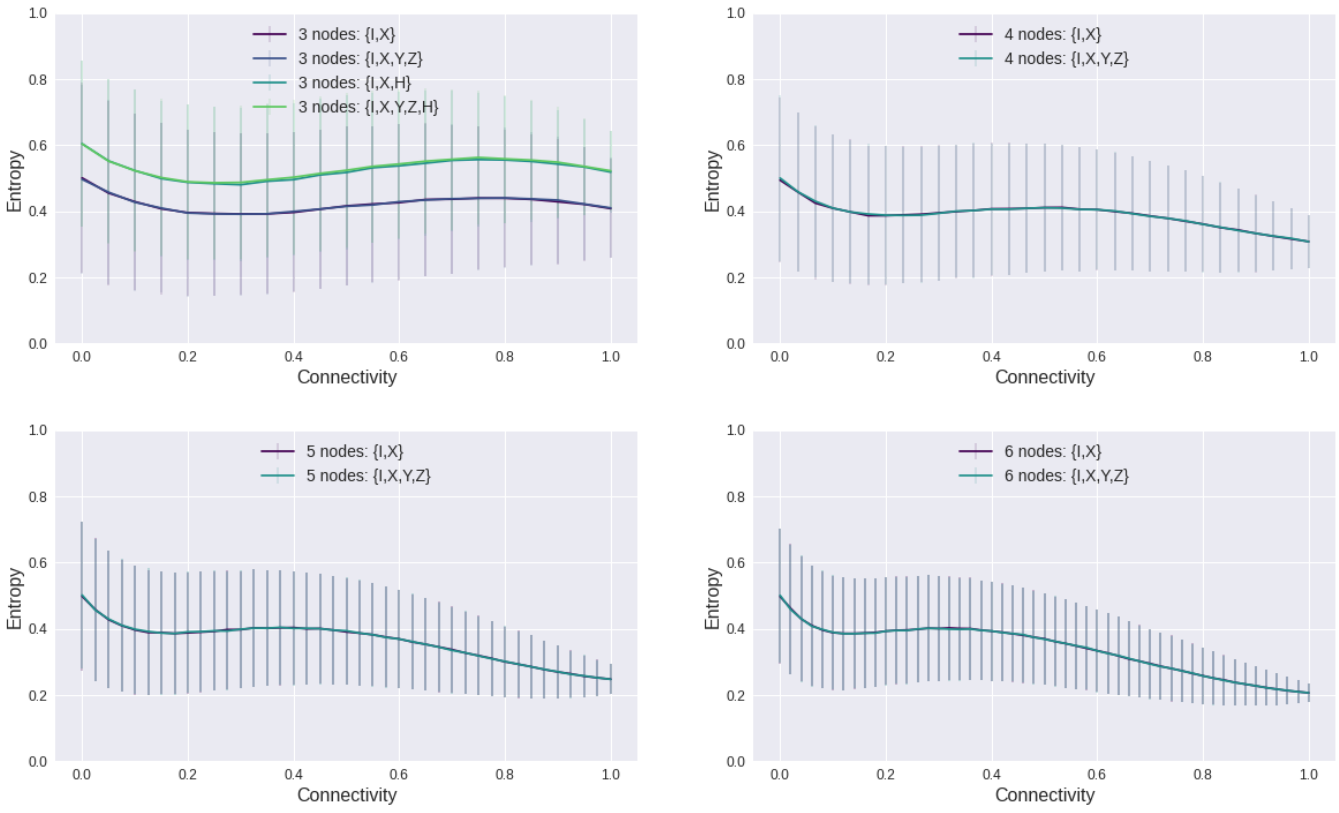


Figure A3. Entropy for QBNs with 3, 4, 5, and 6 nodes. Connectivity is normalized from the range $(1, N)$. The entropy was computed from 20,000 realizations. For practical reasons (extremely large cycles and computational errors), configurations that can produce superpositions were omitted.

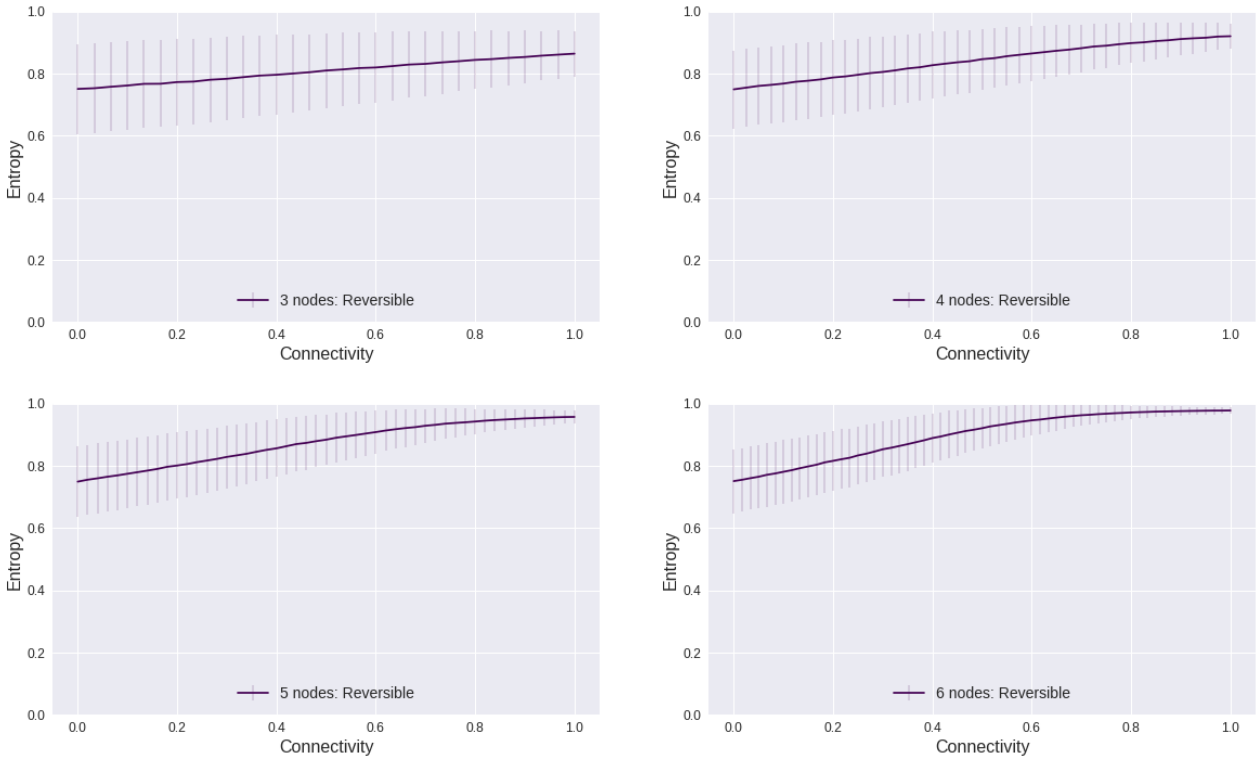


Figure A4. Entropy for RevBNs with 3, 4, 5, and 6 nodes. Connectivity is normalized from the range $(1, N)$. The entropy was computed from 20,000 realizations.

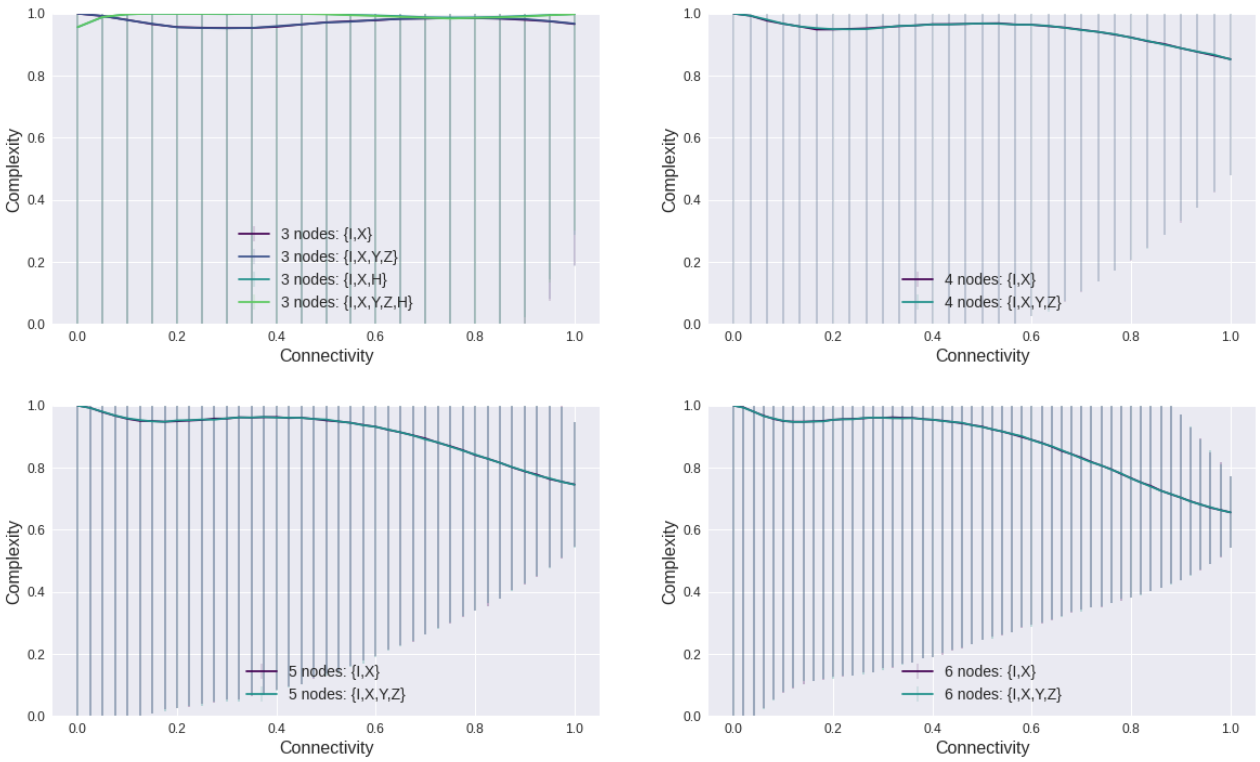


Figure A5. Complexity for QBNs with 3, 4, 5, and 6 nodes. Connectivity is normalized from the range $(1, N)$. The complexity was computed from 20,000 realizations.

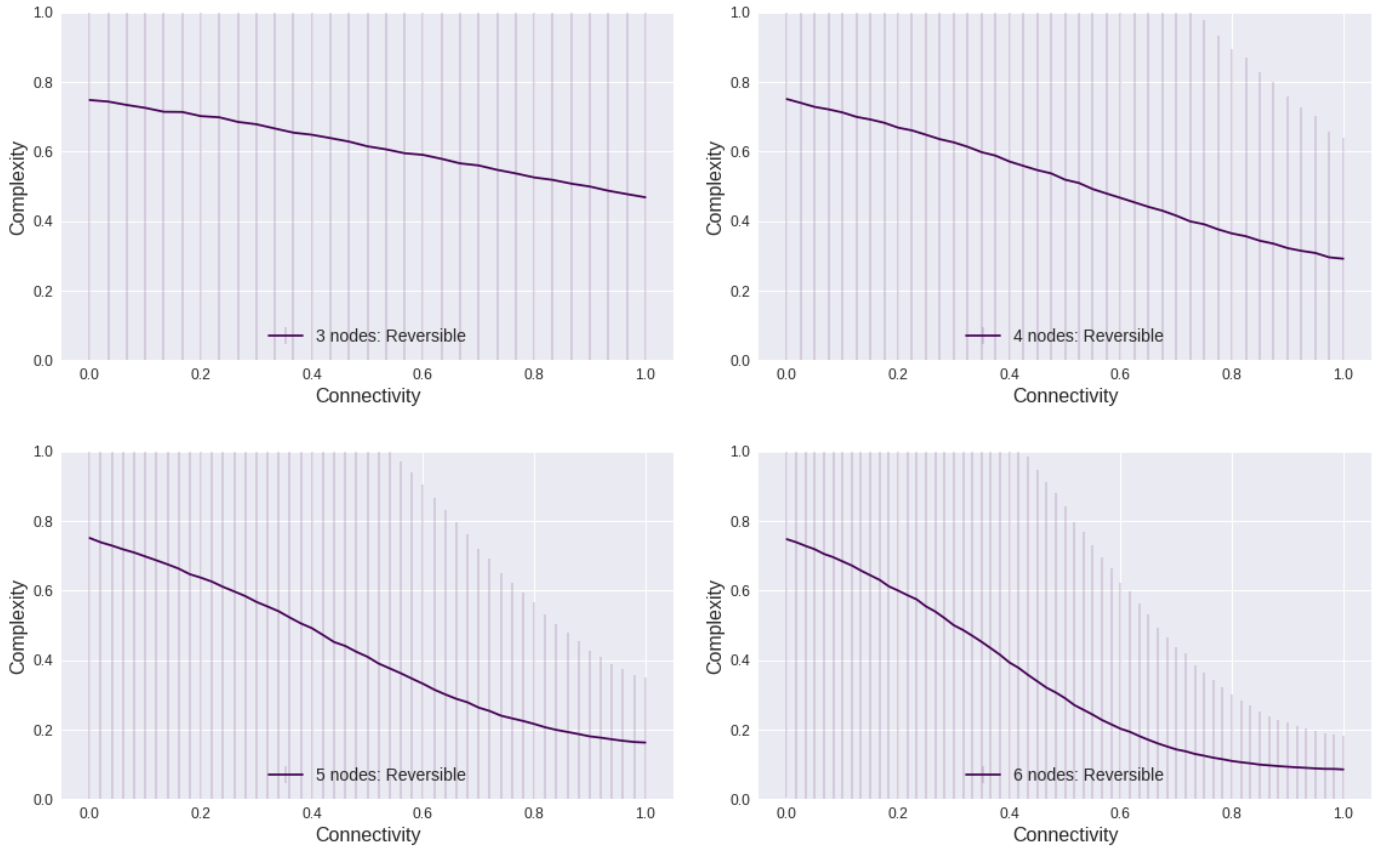


Figure A6. Complexity for RevBNs with 3, 4, 5, and 6 nodes. Connectivity is normalized from the range $(1, N)$. The complexity was computed from 20,000 realizations.

References

1. Kauffman, S.A. Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.* **1969**, *22*, 437–467. [CrossRef]
2. Benioff, P. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *J. Stat. Phys.* **1980**, *22*, 563–591. [CrossRef]
3. Isham, C.J. *Lectures on Quantum Theory: Mathematical and Structural Foundations*; World Scientific: Singapore, 1995.
4. Landauer, R. Irreversibility and heat generation in the computing process. *IBM J. Res. Dev.* **1961**, *5*, 183–191. [CrossRef]
5. Morita, K. *Theory of Reversible Computing*; Springer: Berlin, Germany, 2017.
6. Frank, M.P. Approaching the physical limits of computing. In Proceedings of the 35th International Symposium on Multiple-Valued Logic (ISMVL'05), Calgary, BC, Canada, 19–21 May 2005; pp. 168–185.
7. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information: 10th Anniversary Edition*; Cambridge University Press: Cambridge, UK, 2010.
8. Gershenson, C. Introduction to random Boolean networks. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems*; MIT Press: Cambridge, MA, USA, 2004; pp. 160–173.
9. Gershenson, C. Classification of random Boolean networks. In *Proceedings of the Eighth International Conference on Artificial Life*; MIT Press: Cambridge, MA, USA, 2002; pp. 1–8.
10. Gershenson, C.; Broekaert, J.; Aerts, D. Contextual random Boolean networks. In *European Conference on Artificial Life*; Springer: Berlin, Germany, 2003; pp. 615–624.
11. Shmulevich, I.; Dougherty, E.R.; Kim, S.; Zhang, W. Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics* **2002**, *18*, 261–274. [CrossRef] [PubMed]
12. Zapata, O.; Gershenson, C. Random fuzzy networks. In *Artificial Life Conference Proceedings 14*; MIT Press: Cambridge, MA, USA, 2014, pp. 427–428.
13. Coppersmith, S.; Kadanoff, L.P.; Zhang, Z. Reversible Boolean networks I: distribution of cycle lengths. *Phys. D Nonlinear Phenom.* **2001**, *149*, 11–29. [CrossRef]

14. Coppersmith, S.; Kadanoff, L.P.; Zhang, Z. Reversible Boolean networks: II. Phase transitions, oscillations, and local structures. *Phys. D Nonlinear Phenom.* **2001**, *157*, 54–74. [CrossRef]
15. Montanaro, A.; Osborne, T.J. Quantum boolean functions. *Chic. J. Theor. Comput. Sci.* **2010**, *1*, 1–45.
16. Jones, T.; Brown, A.; Bush, I.; Benjamin, S.C. QuEST and high performance simulation of quantum computers. *Sci. Rep.* **2019**, *9*, 1–11. [CrossRef] [PubMed]
17. Fernández, N.; Maldonado, C.; Gershenson, C. Information measures of complexity, emergence, self-organization, homeostasis, and autopoiesis. In *Guided Self-Organization: Inception*; Springer: Berlin, Germany, 2014; pp. 19–51.
18. Gershenson, C.; Fernández, N. Complexity and information: Measuring emergence, self-organization, and homeostasis at multiple scales. *Complexity* **2012**, *18*, 29–44. [CrossRef]