Noname manuscript No.
(will be inserted by the editor)

Analysis of Boolean functions based on Interaction graphs and their influence in System Biology

Ranjeet Kumar Rout · Santi P. Maity · Pabitra Pal Choudhury · Jayanta Kumar Das · Sk. Sarif Hassan · Hari Mohan Pandey

Received: date / Accepted: date

Abstract Biological regulatory network can be modeled through a set of Boolean functions. These set of functions enable graph-representation of the network structure and hence the dynamics of the network can be seen easily. In this article, the regulations of such network have been explored in term of interaction graph. With the help of Boolean function decomposition this work presents an approach for construction of interaction graphs. This decomposition technique is also used to reduce the network state space of the cell cycle network Fission Yeast for finding the singleton

Ranjeet Kumar Rout
Department of Computer Science and Engineering,
National Institute of Technology, Srinagar,
Hazratbal, Kashmir-190006, India
E-mail: ranjeetkumarrout@nitsri.net

Santi P. Maity
Department of Information Technology,
Indian Institute of Engineering Science and Technology,
Botanical Garden, Shibpur, Howrah, India-711103.

Pabitra Pal Choudhury
Applied Statistics Unit
Indian Statistical Institute, 203 B. T. Road
Kolkata-700108, West Bengal, India.

Jayanta Kumar Das
Applied Statistics Unit
Indian Statistical Institute, 203 B. T. Road
Kolkata-700108, West Bengal, India.

Sk. Sarif Hassan
Department of Mathematics, Pingla Thana Mahavidyalaya
Vidyasagar University Maligram, West
Midnapore-721140, West Bengal, India.

Hari Mohan Pandey
Department of Computer science
Edge Hill University, Ormskirk, Lancashire, UK

attractors. Some special classes of Boolean functions with respect to the interaction graphs have been discussed. A unique recursive procedure is devised that uses the Cartesian product of sets starting from the set of one variable Boolean function. Interaction graphs generated with these Boolean functions have only positive/negative edges and the corresponding state spaces have periodic attractors with length one/two.

Keywords Boolean functions · Boolean networks · Interaction graphs · Singleton attractors · Classification.

1 Introduction

A large amount of experimental data has influenced the development of innovative computational techniques for modeling biological networks. There are various mathematical models, such as Bayesian networks, probabilistic Boolean networks and networks involving ordinary differential equations are used in modeling of biological networks. The ordinary differential equations [4,38,6,28,39,15] are not much used since the approach is deterministic and thus offer average behavior. It is worth mentioning at this point that among all the models, the Boolean network (BN) [50–52] has received much attention due to its simplicity and potential to modeling large number of nodes within a network. This model is used to study the interactions among genes/proteins. Gene expression is a complex process regulated at several stages in the synthesis of proteins. The proteins fulfilling the regulatory functions are produced by other genes for which the genetic regulatory system [46,40,13] is structured by networks of regulatory interactions among deoxyribonucleic acid (DNA), ribonucleic acid (RNA), proteins and small molecules. BN model is very simple but its dynamic process is complex and can yield insight to global behavior of large genetic regulatory network. Complexity of the BN model can be studied with the help of the various properties of Interaction Graph (IG) and its associated Boolean functions [31]. The biochemical reactions are used to form the network that control the Fission Yeast cell-cycle, have been discussed in detail over the last decades [29,5]. Some other models of the genetic network underlying flower development in [14], other cell-cycle networks in [24], and Chinese Hamster Ovary (CHO) cells in [32] have been also studied. Stability, bifurcation and periodic oscillation of two-gene regulatory networks mediated by small RNAs (sRNAs) with multiple delays has been studied [47,48]. A particular set of Boolean functions with certain interaction signs (+/-) of a interaction graph (IG) can provide various insights in the study of dynamical behavior of such biological networks. Thus, the main objective of this work is to find the results of the co-relation between the IG and the BNs state space, which will throw light in modeling the regulatory networks with the most reliable information in biological system.

1.1 Related Review

The BN was originally introduced by Kauffman [22,16]. The use of BNs in the study and analysis of System Biology has been elaborately discussed in [21,25,42,41].

The total number of possible states for a BN with n genes is 2^n. One of the main problems in BNs for use in biological or any physical dynamics is the identification of the updating rules on using observed data [9]. An algorithm for constructing a Probabilistic BNs (PBNs) using Markov chain process and the transition probability matrix have been discussed in [10]. A method has been proposed in [43] to describe a pattern for gene network inference from microarray data and some prior biological knowledge. In a BN, gene expression states are characterized to only two levels: 1 (expressed) and 0 (unexpressed). Although such binary expression is very simple, it retains meaningful biological information contained in real continuous-domain in gene expression profile [19]. The state of target gene is determined by the state of its regulating genes (input genes) and its Boolean functions [20]. However, for the initial condition, system would eventually evolve into a set of stable states called attractors. The set of states that can lead the system to specific attractors is called basin of attraction [30] and the attractors in particular are the most important characteristics because they are mostly related to some specific physiological responses in biological networks [18,2,12,8,49]. An attractor having only one state is called a singleton attractor or fixed point, otherwise it is called a cyclic attractor.

Exploring the BN states spaces would be too time consuming, even for small n, since 2^n states have to be examined to get the singleton attractors. In order to find a singleton attractor, a lot of trajectories may have to be examined. Authors have proved that finding a singleton attractor is NP-hard [1]. To identify BNs which are biologically relevant is a major problem as the number of Boolean functions and the size of the state space of BNs are growing exponentially [31] with the increase in the number of components. An alternate approach for static analysis of BNs through IG has been studied in [31]. In an IG, the nodes/genes are regulated by each with a single Boolean function and types of the functions reflex the sign interactions among the nodes. Interaction strength between genes has been studied using various methods like signaling pathway impact analysis (SPIA) methods, SPIA based on Pearson correlation coefficient (PSPIA), and mutual information (MSPIA) in [3].

1.2 Scope and contribution of the work

Each BN has some biological components which are independently represented by local logical Boolean functions and associate with a Boolean value for each component in BNs. All Boolean functions are not accurately reflecting the behaviors of Biological systems. A subset of Boolean functions having novel characteristics of dynamics of BNs is constructed. One such notable class and their biological properties have been introduced by Kauffman [22]. To identify BNs which are biologically relevant is a major problem. The IG helps us to get some static analysis of BN, a class of Boolean functions play significant role in IG towards the dynamics of BNs state space. Thus, it is imperative to recognize classes of Boolean functions with biologically relevant properties based on sign interactions, which is one of the major objectives of this article. It may not be out of context to mention that the work proposed in [45,33] used for the representation of Boolean functions in terms of Jacobian matrix that has been used to get the local regulatory network. In the proposed work,

we have considered the direct relationship between the variables ((n-1) numbers) and a single function where we used two bits of binary values of the function, the positive and negative interactions are defined and accordingly Boolean functions are classified. Subsequently, a recursive method to generate functions of the higher variables on same characteristics is also suggested. The work reported in [44,36] is used to reduce the number of nodes with respect to variables by considering the Boolean expression. In this proposed work, the network state space reduction depends upon Boolean functions (Truth Table) recursively i.e. from n-variable to $(n-1)$-variable. So, the state space is reduced from 2^n to $2^{(n-1)}$, this offers less number of search space.

The singleton attractors correspond to steady states in BNs and have close relation with steady states in other mathematical models of biological networks [27].The dynamical properties of BNs are analyzed mainly by finding steady-states, attractors and the size of each attractor. Therefore it is meaningful to identify singleton attractors of a given BN. Although it is not plausible that the singleton attractor problem can be solved efficiently (i.e., polynomial time) in all cases, it may be possible to develop algorithms that are faster in practice and/or in the average case. There are some other methods that have been proposed in BN model to reduce the BNs state space using Boolean expression [44,36,53,54]. So the other objective of this work is to provide efficient algorithms to study the properties of BNs as well as biological networks with respect to singleton attractors. For the sake of simplicity toy networks are considered for representation and analysis. An algorithm is augmented in recursive manner to identify the singleton attractors. The work reported in [24,11,7,26] considered various biological networks which have been modeled through BNs having different variables. The works also propose various algorithms to identify large number of singleton attractors. The present work makes use of their result for modeling Biological networks. Here two theorems (Theorem 3 and Theorem 4) are proposed and are proved with the help of toy networks. Theorem 3 identifies BNs having large number of singleton attractors whereas Theorem 4 is used to identify BNs having more number of attractors with length two. This way the theorems (Theorem 3 and Theorem 4) establish a link towards modeling complex Biological network using the toy networks.

In brief, the contributions of this work are summarized as follows:

(1) *Formulation of sign (+/-) interaction graph using Boolean function decomposition* - An n-variable Boolean function is required to be decomposed into 2^{n-1} segments taking $(n-1)$-variables at a time having bit length two. So the possible output of each segmented Boolean function is a two bits string $(00,01,10,11)$. These two bit segments are essential for construction of signed IGs.
(2) *Recursive procedure for obtaining Boolean functions* - Identification of biologically relevant Boolean functions are possible from 1-variable to n-variable recursively. A unique recursive procedure is devised that uses the Cartesian product of sets starting from the set of one variable Boolean functions. Interaction graphs generated with these Boolean functions have only positive/negative edges and the corresponding BN state spaces have periodic attractors with length one/two.

(3) *Reduction of Boolean network state space of cell cycle network of Fission Yeast* - The decomposition technique is also used to reduce the network state space of the cell cycle network Fission Yeast for finding the singleton attractors. In the next sections these points are discussed briefly.

The remainder of the paper is organized as follows: In Section 2, definition and notation of IG, Boolean function segmentation procedure is discussed. Section 3 presents a recursive procedure to obtain Boolean functions responsible for IG having only positive/negative edges. Some significant properties of these Boolean functions are also discussed in this section. In Section 4, a state transition model of cell cycle network of Fission Yeast is taken and its network state space reduction procedure has been discussed. Section 5 deals with concluding remarks emphasizing the key factors of the entire analysis.

2 Definition and Notations

An n-variable Boolean function f is a mapping from the set of all possible n-bit strings $\{0,1\}^n$ into $\{0,1\}$. The number of different n-variable Boolean functions is 2^{2^n}, where each function can be represented by a truth table output as a binary string of length 2^n. The decimal equivalent of the binary string starting from bottom to top (least significant bit) in the truth table is called the rule number of that function [37]. The complement of a Boolean function f is denoted as $\bar{f}$.
An example of Boolean function in truth table and its complement having rule number is given in Table 1. Note that the functions represented in the Table 1 topmost row are expressed as *Algebraic Normal Forms* (ANF) [37].
An n-variable Boolean function can be decomposed into 2^{n-1} segments taking $(n-1)$-variable at a time having bit length two. So the possible output of each segmented Boolean function is a two bits string $(00,01,10,11)$. Decomposition technique of an n-variable Boolean function is discussed in the subsection 2.1:

Table 1 Example of Truth Table,(for $n = 3$).

x_1	x_2	x_3	$f_1(x) = x_1 \oplus x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus 1$	$\bar{f}_1(x) = x_3 \oplus x_1x_2 \oplus x_1x_2x_3$
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

The truth table of a Boolean function deciphers various important properties including *Hamming weight* and *Hamming distance* of Boolean functions.

Definition 1 The *Hamming weight* of a Boolean function $f \in F^n$ is defined as the number of non-zero values in the binary truth table, which is denoted as $w(f)$.

For example, the Hamming weight $w(f = x_3) = w((10101010)_2) = 4$.

Here, F^n denotes the set of all possible $n-$variables Boolean functions. For simplicity, in the rest part of the paper F is taken as a set of Boolean functions such that $F \in F^n$.

Definition 2 The *Hamming distance* between two functions $f \in F^n$ and $g \in F^n$ is defined as the number of truth table positions in which the functions differ. Thus, it is the XOR sum of the two functions $d(f,g) = w(f \oplus g)$.

For example, the Hamming distance between two function $f = x_3$ and $g = x_1x_2x_3$ is $d(f,g) = w(f \oplus g) = w(10101010 \oplus 10000000) = w(00101010) = 3$.

2.1 Decomposition of Boolean functions

In this section, a method has been proposed to decompose a Boolean function which can be expressed with the help of its decomposition fragments through a relation of Boolean decompositions(or expansions) such that each decomposition fragment consists of two bits.

Definition 3 Let $f(x)$ be an output of a Boolean function corresponding to the input vectors x. Let x_1 be one variable of the input vectors x. Now $f(x_1 = 0, x_2, x_3, \ldots, x_n)$ and $f(x_1 = 1, x_2, x_3, \ldots, x_n)$ are two decomposition fragments of the Boolean function f with respect to the constituent variable x_1 of the input vectors x. For simplicity, in the rest part of the paper, $f(x_1 = 0, x_2, x_3, \ldots, x_n)$ as f_0^1 and $f(x_1 = 1, x_2, x_3, \ldots, x_n)$ as f_1^1 are denoted.

In the similar fashion, for the two constituent variables x_1 and x_2 of the input vectors x for a Boolean function f of $n-$variables, the decomposition fragments are $f(x_1 = 0, x_2 = 0, x_3, \ldots, x_n)$, $f(x_1 = 0, x_2 = 1, x_3, \ldots, x_n)$, $f(x_1 = 1, x_2 = 0, x_3, \ldots, x_n)$, $f(x_1 = 1, x_2 = 1, x_3, \ldots, x_n)$ and the decomposition fragments are represented as f_{00}^{12}, f_{01}^{12}, f_{10}^{12} and f_{11}^{12}, respectively.

Proposition 1 *The decomposition of a n-variable Boolean function with respect to the* $(n-1)$ *variable generates* 2^{n-1} *decomposition fragments and the length of each decomposition fragment is* $2^{n-(n-1)}$.

Decomposition of any Boolean function f with respect to an arbitrary input variable x_i have two segments f_0^i and f_1^i, where $i \in \{1, 2, 3, \ldots, n\}$, which is defined in (1)

$$D_i(f) = \left\{ \begin{array}{l} f_0^i = f(x_1, x_2, x_3, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \\ f_1^i = f(x_1, x_2, x_3, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) \end{array} \right\} \tag{1}$$

The bit string representations of f_0^i and f_1^i are called decomposition fragments of the Boolean function f having each 2^{n-1} number of bits in length.

In the similar fashion, the decomposition of an $n-$variable Boolean function f with respect to any two arbitrary input variables x_i and x_j have four segments f_{00}^{ij}, f_{01}^{ij}, f_{10}^{ij} and f_{11}^{ij}, where $i, j \in \{1,2,3,\ldots,n\}$, which are defined in (2)

$$D_{ij}(f) = \left\{ \begin{array}{l} f_{00}^{ij} = f(x_1,x_2,x_3,\ldots,x_{i-1},0,x_{i+1},\ldots,x_{j-1},0,x_{j+1},\ldots x_n) \\ f_{01}^{ij} = f(x_1,x_2,x_3,\ldots,x_{i-1},0,x_{i+1},\ldots,x_{j-1},1,x_{j+1},\ldots x_n) \\ f_{10}^{ij} = f(x_1,x_2,x_3,\ldots,x_{i-1},1,x_{i+1},\ldots,x_{j-1},0,x_{j+1},\ldots x_n) \\ f_{11}^{ij} = f(x_1,x_2,x_3,\ldots,x_{i-1},1,x_{i+1},\ldots,x_{j-1},1,x_{j+1},\ldots x_n) \end{array} \right\} \quad (2)$$

In general, the decomposition of an n-variable Boolean function f with respect to any arbitrary $(n-1)$ constituent variables $x_{i_1}, x_{i_2}, \ldots, x_{i_{n-1}}$ generate 2^{n-1} segments $f_{(0,0,\ldots,\ldots,\ldots,0)}^{i_1,i_2,\ldots,i_{n-1}}, \ldots, f_{(1,1,\ldots,\ldots,\ldots,1)}^{i_1,i_2,\ldots,i_{n-1}}$ which are defined in (3) as follow;

$$D_{i_1,i_2,\ldots,i_{n-1}}(f) = \left\{ \begin{array}{l} f_{0,\ldots\ldots\ldots,0}^{i_1,i_2,\ldots,i_{n-1}} = f(x_1,\ldots,x_{i_1-1},0,x_{i_1+1},\ldots,x_{i_2-1},0,x_{i_2+1},\ldots,x_{i_{(n-1)}-1},0,x_{i_{(n-1)}+1},\ldots x_n) \\ \quad . \\ \quad . \\ \quad . \\ f_{1,\ldots\ldots\ldots,1}^{i_1,i_2,\ldots,i_{n-1}} = f(x_1,\ldots,x_{i_1-1},1,x_{i_1+1},\ldots,x_{i_2-1},1,x_{i_2+1},\ldots,x_{i_{(n-1)}-1},1,x_{i_{(n-1)}+1},\ldots x_n) \end{array} \right\} \quad (3)$$

where $i_1, i_2, \ldots, i_{(n-1)} \in 1,2,3,\ldots,n$.

Illustration:

A 3-variable Boolean function $f(x) = x_1 \oplus x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus 1 = (00010101)_2$ is decomposed by considering 2-variables at a time, to decompose the Boolean function f consequently there are 3 disjoint decomposition fragments $(D_{23}(f), D_{13}(f), D_{12}(f))$ of the function f which are shown below.

$$D_{23}(f(x)) : \left\{ \begin{array}{l} f_{00}^{23} = 11 \\ f_{01}^{23} = 00 \\ f_{10}^{23} = 10 \\ f_{11}^{23} = 00 \end{array} \right\} \; D_{12}(f(x)) : \left\{ \begin{array}{l} f_{00}^{12} = 10 \\ f_{01}^{12} = 10 \\ f_{10}^{12} = 10 \\ f_{11}^{12} = 00 \end{array} \right\}$$

$$D_{13}(f(x)) : \left\{ \begin{array}{l} f_{00}^{13} = 11 \\ f_{01}^{13} = 00 \\ f_{10}^{13} = 10 \\ f_{11}^{13} = 00 \end{array} \right\}$$

Note that the bit-vector 00010101 is a three variable Boolean function, which is represented in the form of truth table as shown in Table 1. Now on wards, any Boolean function will be represented as bit vector only for simplicity. Here $D_{mn}(f)$ indicates decomposition of the function f with respect to variable x_m and x_n.

The definition of IG is based on the decomposition technique and the analysis of an IG can be performed with the help of decomposition fragments of any Boolean function.

2.2 Boolean Network

A *BN* is a discrete dynamical system that consists of n Boolean components which are mutually interacting in a discrete time stamp. Dynamics of such a system is usually described by a directed graph called interaction graph. Several IGs can be considered, in the scope of this paper we focus on the synchronous and generalized iteration graphs [31].

2.3 Interaction Graph (IG) of F

An $IG = (V,E)$ consists of a set of n vertices $V = \{v_1, v_2, \ldots, v_n\}$, each vertex is associated with a Boolean function, and set of edges $(v_i, v_j) \in E$, each edge is associated with a sign $\{+/-\}$ directed edge from v_i vertex to v_j vertex.There exists an arc $v_i \to \{+/-\}$, $v_j \in E$, *if and only if* there exist at least one 01 or 10 in decomposition fragments of the representative function (Discussed in Illustration given below) for positive and negative arc, respectively. For modeling of Biological network, in the framework of IG, the vertices represent genes and the corresponding associated Boolean functions perceive the gene regulatory rules.

Illustration: For $n = 3$ and F is defined by:

$$F = \begin{cases} f_1(x) = x_1x_2x_3, \\ f_2(x) = x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3, \\ f_3(x) = x_1x_2 \oplus x_2x_3 \oplus x_1x_2x_3. \end{cases}$$

the corresponding decomposition fragments of the three functions $f_1(x), f_2(x)$ and $f_3(x)$ are shown below.

$$D_{23}(f_1(x)/f_2(x)/f_3(x)) : \left\{ \begin{array}{l} f_{00}^{23} = 00/00/00 \\ f_{01}^{23} = 00/01/00 \\ f_{10}^{23} = 00/00/01 \\ f_{11}^{23} = 01/11/11 \end{array} \right\}$$

$$D_{13}(f_1(x)/f_2(x)/f_3(x)) : \left\{ \begin{array}{l} f_{00}^{13} = 00/00/00 \\ f_{01}^{13} = 00/01/01 \\ f_{10}^{13} = 00/00/01 \\ f_{11}^{13} = 01/11/01 \end{array} \right\}$$

$$D_{12}(f_1(x)/f_2(x)/f_3(x)) : \left\{ \begin{array}{l} f_{00}^{12} = 00/00/00 \\ f_{01}^{12} = 00/01/01 \\ f_{10}^{12} = 00/01/00 \\ f_{11}^{12} = 01/01/11 \end{array} \right\}$$

For each Boolean function stated above, there are three decomposition segments. So there are total $3^2 = 9$ decomposition segments is discussed in proposition 4. Output of decomposition segments (first segment for function $f_1(x)$, second segment for function $f_2(x)$ and third segment is for $f_3(x)$ are shown above.

Representation of edge connectivity of these three functions as per the definition of IGs are shown in Fig.1(a), where node 1, 2 and 3 are regulated/interacted through

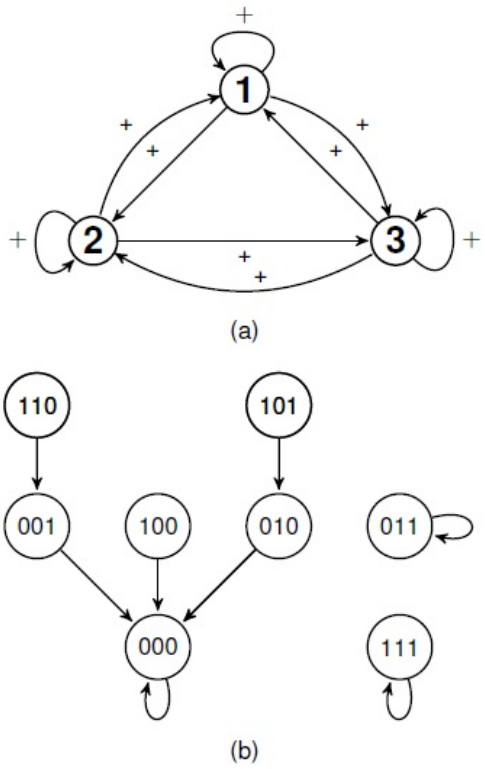

Fig. 1 (a) IG of the Boolean functions $f_1(x), f_2(x)$ *and* $f_3(x)$. and (b) BN state space of the Boolean function $f_1(x), f_2(x)$ *and* $f_3(x)$.

the functions f_1, f_2 and f_3, respectively and the corresponding Boolean network state space having three singleton attractors $(000 \longrightarrow 000), (011 \longrightarrow 011)$ and $(111 \longrightarrow 111)$ are shown in Fig.1(b)

3 Recursive Procedure to obtain Boolean functions responsible for IGs having positive edges only

Let $S_1 = \{00, 10, 11\}$ be a set of 1-variable Boolean functions. Here, it is seen from the very definition of the interaction graph in section 2.3, that all the Boolean functions in the set S_1 are responsible for producing IG having positive edges only. The complement functions of the set S_1 are responsible for generating IG having negative edges only and the complement set is $\bar{S_1} = \{11, 01, 00\}$. Note that the Boolean functions $\{00, 11\}$ are present in both the cases, which are constant and the same convention needs to be followed for any arbitrary $n-$variable Boolean functions.

Let f and g be two Boolean functions of S_1. The concatenation [35][34] of the Boolean function f with itself and the concatenation of f with g are defined as ff and fg, respectively. For example $f = 01$ and $g = 11$, then $ff = f \times f = 01 \times 01 = 0101$ and $fg = f \times g = 01 \times 11 = 0111$. And the hamming weight of the function f and g are denoted as$(w(f))$ and $(w(g))$, where $w(f) = 1$ and $w(g) = 2$.
Note that, if f and g are the Boolean functions of n-variable then ff and fg are Boolean functions of $(n+1)$-variable. So the Cartesian product with concatenation operation of the set S_1 with itself is defined successively as follows:

$S_1 \times S_1 = \{\{0000\}, \{1000, 0010\}, \{1010, 1100, 0011\}, \{1110, 1011\}, \{1111\}\}$.

Here, S_1 contains three classes each with a 1-variable Boolean function having hamming weight 0,1 and 2 respectively, whereas $S_1 \times S_1$ set contains five disjoint classes of 2-variable Boolean functions having hamming weight $0, 1, 2, 3$ and 4 with cardinality 1, 2, 3, 2 and 1 for the class 0, class 1, class 2, class 3 and class 4 respectively. This process is repeated for next higher variable using the recursive formula of the following:

Base Case (for $n = 1$)

$$S_1 = \{00, 10, 11\} \textit{ and } f_1 \in S_1 \tag{4}$$

Recursion (for $n \geq 2$)

The concatenation of two Boolean functions is defined as

$$f_n = f_{(n-1)} f_{(n-1)} \textit{ or } f_{(n-1)} g_{(n-1)} | w(f_{(n-1)}) > w(g_{(n-1)}) \vee (f_{(n-1)} = g_{(n-1)}) \quad \textit{where } f_{(n-1)}, g_{(n-1)} \in S_{(n-1)} \tag{5}$$

Therefore, recursive Cartesian product is defined as $S_n = S_{(n-1)} \times S_{(n-1)}$, where S_n contains n-variable Boolean functions having classes-$0, 1, 2, \ldots, 2^n$ with Hamming weight $0, 1, 2, \ldots, 2^n$ respectively.

Theorem 1 *The recursive procedure as stated in* (4) *and* (5)*, is applied* $(n-1)$*-variable, the recursion generate a set of n-variables Boolean functions, which are responsible for IGs having positive edges only.*

Proof: When $n = 1$, that is for the base case of the recursion, the least significant bit (LSB) position of all the Boolean functions in the set $S_1 = \{00, 10, 11\}$ is either less or same Hamming weight with respect to the most significant bit (MSB). The Cartesian product and Concatenation operation on the set $S_1 = \{00, 10, 11\}$ is shown in Fig.2 for generating 2-variable Boolean function.

Here, S_1 contains three Boolean functions of 1-variable and the set $S_1 \times S_1$ contains nine $2-$variable Boolean functions. Red colored Boolean functions are not responsible for generating IG having all edges are positive in case of 2-variable. Any n-variable Boolean function can be decomposed into two $(n-1)-$variable Boolean functions. As per the recursive formula defined in the expression (4) and (5), only six functions are responsible for generating the graph having all positive edges and all are distributed among the Hamming weight $0, 1, 2, \ldots, 2^n$. Therefore, the recursive procedure using the Cartesian product also preserves the same property for the next higher variable.

Illustration:

$$S_2 = S_1 \times S_1 = \left\{ \begin{array}{c} 00 \\ 10 \\ 11 \end{array} \right\} \times \left\{ \begin{array}{c} 00 \\ 10 \\ 11 \end{array} \right\} = \left\{ \begin{array}{c} 0000 \\ 0010 \\ 0011 \\ 1000 \\ 1010 \\ 1011 \\ 1100 \\ 1110 \\ 1111 \end{array} \right\}$$

Fig. 2 Shows 2-variable Boolean functions, generated recursively from 1−variable Boolean function.

The set of 2−variables Boolean functions which produce IGs having positive edges is $S_2 = \{0000, 1000, 1010, 1100, 1110, 1111\}$. Now, in the similar way, the 3-variables Boolean functions which generate IGs can be obtained by the Cartesian product of $S_2 \times S_2 = S_3$ (say). Only twenty number of Boolean functions are satisfied the recursive formula defined in (5). The Boolean functions are shown in Fig.3 and Boolean functions from 2 to 4-variable are shown in Appendix A.

$$S_3 = S_2 \times S_2 = \left\{ \begin{array}{c} 0000 \\ 1000 \\ 1010 \\ 1100 \\ 1110 \\ 1111 \end{array} \right\} \times \left\{ \begin{array}{c} 0000 \\ 1000 \\ 1010 \\ 1100 \\ 1110 \\ 1111 \end{array} \right\} = \left\{ \begin{array}{c} 00000000 \\ 10000000 \\ 10001000 \\ 10100000 \\ 11000000 \\ 10101000 \\ 11001000 \\ 11100000 \\ 10101010 \\ 11001100 \\ 11101000 \\ 11110000 \\ 11101010 \\ 11101100 \\ 11111000 \\ 11101110 \\ 11111010 \\ 11111100 \\ 11111110 \\ 11111111 \end{array} \right\}$$

Fig. 3 Shows 3−variable Boolean functions, recursively generated from 2−variable Boolean function.

Theorem 2 *The number of subsets of S_n for n-variable Boolean function is* $2^n + 1$.

Proof: We now prove the theorem using principle of *Mathematical Induction*. For $n = 1$, $S_1 = \{00, 10, 11\}$ contains three Boolean functions and there are three classes having Boolean functions of Hamming weight 0,1,and 2. Consequently, the base case is true.
By using the recursion equation (1) and (2), when $n = 2$ the set is as follows $S_2 = \{\{0000\}, \{1000\}, \{1010, 1100\}, \{1110\}, \{1111\}\}$, which contains six Boolean functions having Hamming weight $0, 1, 2, 3$, and 4. Hence for $n = 2$, cardinality of S_2 is $2 \times 2 + 1 = 2^2 + 1 = 5$.

Induction Hypothesis:

Assume that, the cardinality of S_k is $2^k + 1$. Thus by induction hypothesis, the number of classes of k-variable is calculated by using the formula as given below.

$$S_k = 2 \times 2 \times 2 \times \cdots \times 2(k - times) + 1 = 2^k + 1 \tag{6}$$

Induction: Here we have to prove that the result is true for $(k+1)$-variable. So, the function is $S_{k+1} = 2 \times 2 \times \cdots \times 2(k + 1 - times) + 1 = 2 \times (2 \times 2 \times \cdots \times 2(k - times)) + 1 = 2 \times 2^k + 1 = 2^{(k+1)} + 1$. So, the formula is true for all the values of $k = 1, 2, 3, \ldots, n$.
Hence, by the principle of mathematical induction, we conclude that S_n is true for all positive integers n. The naming of the classes are given as class 0, class 1, ..., class 2^{n+1} for n-variable.

Corollary 1 *The number of functions for n-variable is calculated by the following recurrence relation as given below:*

$$\rho = \sum_{i=0}^{2^n} f^n(i)$$

where

$$f^n(i) = \left\{ \begin{array}{l} f^{n-1}(\frac{i}{2}) + \sum\limits_{k=1}^{2^n-1} \sum\limits_{j=0}^{k-1} (f^{n-1}(k) \times f^{n-1}(j)), \\ for\, i = 0, 2, \ldots, 2^n \, and \, i = k + j \\ \sum\limits_{k=1}^{2^n-1} \sum\limits_{j=0}^{k-1} (f^{n-1}(k) \times f^{n-1}(j)), \\ for\, i = 1, 3, \ldots, 2^n \, and \, i = k + j \end{array} \right\}$$

where $f^n(i)$ is the number of n-variable Boolean functions having Hamming weight i and the number of Boolean functions from 1 to 5 variable is discussed in Illustration 4.

Illustration: For 1-variable Boolean functions, the set $S_1 = \{00, 10, 11\}$ is used for generating IG having positive edges only, here the cardinality is three i.e. $f^1(0) = 1$, $f^1(1) = 1$ and $f^1(2) = 1$ which is the base case. For $n = 2$, the set $S_2 = \{\{0000\}, \{1000\}, \{1010, 1100\}, \{1110\}, \{1111\}\}$ and the cardinality is six. From the above equation $\rho = \sum\limits_{i=0}^{2^2} f^2(i) = f^2(0) + f^2(1) + f^2(2) + f^2(3) + f^2(4)$. Here $f^2(0) = f^1(0) = 1$, $f^2(1) = f^1(0) \times f^1(1) = 1 \times 1 = 2$, $f^2(2) = f^1(2) + (f^1(2) \times f^1(0)) = 1 + (1 \times 1) = 2$, $f^2(3) = f^1(2) \times (f^1(1) = 1 \times 1 = 1$ and $f^2(4) = f^1(\frac{4}{2}) = f^1(2) = 1 \times 1 = 1$.

Hence the total number of functions $\rho = \sum_{i=0}^{2^2} f^2(i) = f^2(0) + f^2(1) + f^2(2) + f^2(3) + f^2(4) = 1+1+2+1+1 = 6$ are responsible for generation of IG having positive edges only. Similarly, $\rho = 20, \rho = 192, \rho = 16860$ for $n = 3,4$ *and* 5 respectively.

Theorem 3 *If $IG(F)$ has only positive edges, then the number of singleton attractors $(p = 1)$ (cycle length one) are more than the number of p-adic attractors where $p = 2,3,\ldots,2^n$.*

Proof: For 1-variable Boolean function, the set $S_1 = \{00,10,11\}$ is used for generating IG having positive edges only. The BN state space of n-variable consists of cycle length $1,2,\ldots,2^n$. In case of 1-variable the maximum length of cycle is 2. Here the function $f(x) = x_1 = \{10\}_2$ is used to generate an IG having one positive edge as rest two functions from $S_1 = \{00,11\}$ are neutral as per the definition of the decomposition of Boolean function. The IG and the corresponding BN state space with two singleton attractors for the function $f(x) = x_1$ is shown in Fig.4.

Fig. 4 (a) Interaction Graph of 1-variable Boolean function $f(x) = x_1$ and (b) BN state space of 1-variable Boolean function $f(x) = x_1$.

From the above *BN*s we observe that, there are two cycles having length one and no periodic cycle of length two. In this case, the Hamming weight of the *LSB* bit is either same or less as compared to the *MSB* bit. Here, the vectors in domain have less Hamming weight from top to bottom and as per the truth table representation of Boolean functions, the *LSB* of these Boolean functions are mapped to the top vectors of the domain and *MSB*'s of these Boolean functions are mapped to the bottom vectors of the domain. Hence, single length cycles are generated more as compared to other periodic cycles. By using (2), the set $S_2 = \{\{0000\},\{1000\},\{1010,1100\},\{1110\},\{1111\}\}$ (Truth Table format is in Table 2) is generated using Cartesian product of 1-variable Boolean functions. These Boolean function preserves the same properties, that two bit positions from *MSB* has Hamming weight either same or less with respect to the Hamming weight of two bits from *LSB*. So, the Boolean network state space generated by this type of Boolean functions has large number of singleton attractors as shown in Fig.5(b).

From Table 2, which contains a set of 2-variable Boolean functions F defined as:

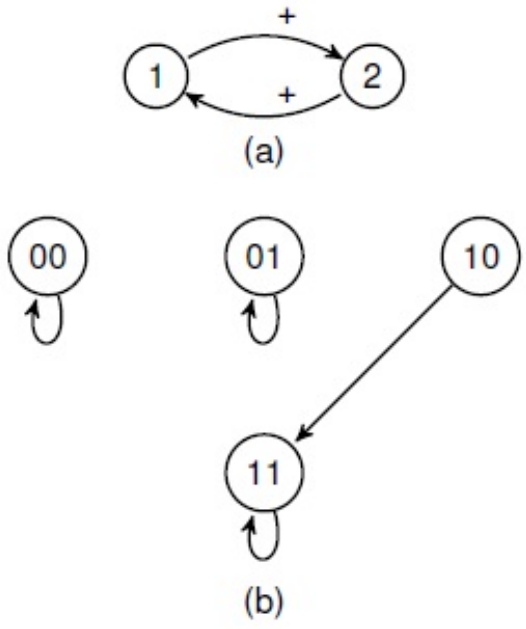

Fig. 5 (a) IG of the Boolean functions $f_1(x) = x_1$ and $f_2(x) = x_1 \oplus x_2 \oplus x_1x_2$ and (b) BN state space of the Boolean functions $f_1(x) = x_1$ and $f_2(x) = x_1 \oplus x_2 \oplus x_1x_2$.

Table 2 Shows Boolean functions responsible for generating BN state spaces having one length cycle.

x_1	x_2	f_1	f_2	f_3	f_4	f_5	f_6
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	1
1	0	0	0	0	1	1	1
1	1	0	1	1	1	1	1

$$F = \begin{cases} f_1(x) = 0 \\ f_2(x) = x_1x_2 \\ f_3(x) = x_2 \\ f_4(x) = x_1 \\ f_5(x) = x_1x_2 \oplus x_1 \oplus x_2 \\ f_6(x) = x_1x_2 \oplus x_1 \oplus x_2 \oplus 1 \end{cases}$$

These Boolean functions are used for enumerating IGs having positive edges and the corresponding BN state spaces contains large number of singleton attractors. From the above six functions by taking any two functions the total number of IGs or the corresponding BN state spaces is ${}^nP_k = {}^6P_2 = 30$. Here, we have to show that the average number of one length cycle is more than two length cycle and so on.

The total number of one length cycle is $\psi = \psi_{00} + \psi_{01} + \psi_{10} + \psi_{11}$, where ψ_{00} is the number of one length cycle yielding the vertex "00". Any two functions which maps towards vertex "00" is ${}^5P_2$ as there is only one function in which the *LSB* bit is 1 bit string , whereas rest of the functions (five functions) have bit string 0 in the *LSB* position. Similarly, for the vertex "11", total number of one length cycle is $\psi_{11} = {}^5P_2$. For the vertex "10", total number of one length cycle is $\psi_{10} = 9$ and for the vertex "01", the total number of one length cycle is $\psi_{10} = 9$. So, the average number of one length cycle $\psi_{avg} = \psi/{}^6P_2 = (\psi_{00} + \psi_{01} + \psi_{10} + \psi_{11})/{}^6P_2 = 58/30 = \lceil 1.932 \rceil = 2$.

Here, cycles having length two are not yielding by the vertex "00" and "11". Only by talking f_3 and f_4 in any oder, two 2-length cycle is possible, interestingly that graph contains two 1-length cycle. So the average number of two length cycle is $\psi_{avg}(2) = 2/30 = 0.066$, which is very small in comparison with 1-length cycle. Boolean functions are recursively generated from $n-$variable to $(n+1)-$variable, therefore the recursive procedure using the Cartesian product also preserves the same property for the next higher variable.

The above property is also elaborated using **Markov chain with transition matrix** which is defined as follows:

Let P be a $(2^n \times 2^n)$-matrix with elements $\{P_{i,j} : i, j = 1, 2, \ldots, 2^n\}$. A random process $(X_0, X_1, \ldots, X_{2^n})$ with finite state space $S = \{s_0, s_1 \ldots, s_{2^n}\}$ is said to be a **Markov chain with transition matrix** P [10,23,17]. If for all n, all $i, j \in 1, \ldots, 2^n$ and all $i_0, i_1, \ldots, i_{2^n-1}$, we have $P(X_{n+1} = s_j | X_0 = s_{i_0}, X_1 = s_{i_1}, \ldots, X_{n-1} = s_{i_{n-1}}, X_n = s_{i_n}) = P(X_{n+1} = s_j | X_n = s_i) = P_{i,j}$.

The elements of the transition matrix P are called transition probabilities. The transition probability $P_{i,j}$ is the conditional probability of being in state s_j given that we are in state s_i, where $\{i, j\} \in \{0, 1, \ldots, 2^n\}$ for $n-$variable. The transition matrix or the path matrix for various length of cycles is defined recursively in equation (7) and equation (8):

Base Case (for $k = 1$)

$$\psi^1 = \sum_{i,j=1}^{2^n} P_{i,j}, \forall\ i = j. \tag{7}$$

Recursion (for $k >= 2$) The path matrix having periodic cycle more than one is defined as

$$\begin{aligned} Initialise\ P_{i,j}^{k-1} = 0, \forall\ i = j,\ P_{i,j}^{k} = P_{i,j}^{k-1} \times P_{i,j}^{k-1} \\ \psi^k = \sum_{i,j=1}^{2^n} P_{i,j}, \forall\ i = j. \end{aligned} \tag{8}$$

Illustration: For instance the initial transition matrix $p_{i,j}^1$ generated by all the Boolean functions of 2-variable is

$$p_{i,j}^1 = \begin{bmatrix} 0.66667 & 0.66667 & 0.66667 & 0 \\ 0.2 & 0.3 & 0.3 & 0.2 \\ 0.2 & 0.3 & 0.3 & 0.2 \\ 0 & 0.66667 & 0.66667 & 0.66667 \end{bmatrix}$$

The average number of single length cycle or singleton attractor is

$$\psi^1 = \sum_{i,j=1}^{4} P_{i,j}, \forall\, i = j \text{ i.e. } \psi^1_{avg} = 0.66667 + 0.3 + 0.3 + 0.66667 = 1.93333.$$

Then the average number of cycle having length two will be the matrix multiplication of $p^1_{i,j} \times p^1_{i,j}$ by assigning the diagonal values to zero, which is denoted by $p^2_{i,j}$, and the corresponding matrix is shown below.

$$p^2_{i,j} = \begin{bmatrix} 0 & 0.66667 & 0.66667 & 0 \\ 0.2 & 0 & 0.3 & 0.2 \\ 0.2 & 0.3 & 0 & 0.2 \\ 0 & 0.66667 & 0.66667 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 0.66667 & 0.66667 & 0 \\ 0.2 & 0 & 0.3 & 0.2 \\ 0.2 & 0.3 & 0 & 0.2 \\ 0 & 0.66667 & 0.66667 & 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0.066667 & 0.05 & 0.05 & 0.066667 \\ 0.06 & 0.156667 & 0.156667 & 0.06 \\ 0.06 & 0.156667 & 0.156667 & 0.06 \\ 0.066667 & 0.05 & 0.05 & 0.066667 \end{bmatrix}$$

So, the average number of cycle having length two is $\psi^2 = 0.066667 + 0.156667 + 0.156667 + 0.066667 = 0.446667$. Similarly, by using (8), the cycle length of 3 and 4 is $\psi^3 = 0.0417778$ and $\psi^4 = 0.000570667$ respectively. From the above $\psi^1 > \psi^2 > \psi^3 > \psi^4$. Hence, the average number of one length cycle is large as compared to other periodic cycles. Different average cycle lengths up to 6−variable is shown in Table 3. The bar plot of average number of single ton attractors of these Boolean network state space are also given in Fig.6.

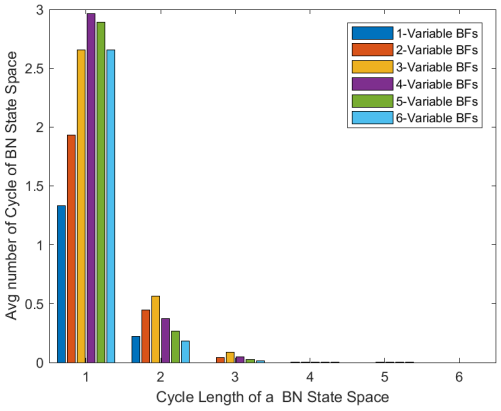

Fig. 6 Bar plot of all Boolean network state space up to 6-variable Boolean function.

Illustration: Let us consider for $n = 3$, the set of Boolean function is defined in F as follows:

Table 3 Shows average number of singleton attractors are more than p-adic attractors in Boolean network state space up to 6-variable.

n	ψ^1	ψ^2	ψ^3	ψ^4	ψ^5	ψ^6	ψ^7	ψ^8	ψ^9	ψ^{10}	ψ^{11}
1	1.333	0.222	-	-	-	-	-	-	-	-	-
2	1.933	0.444	0.0421	0.0005	-	-	-	-	-	-	-
3	2.657	0.565	0.0879	0.0045	$1.300e^{-005}$	$1.07e^{-010}$	$7.36e^{-021}$	$3.47e^{-041}$	-	-	-
4	2.966	0.373	0.0456	0.0014	$1.54e^{-006}$	$1.87e^{-012}$	$2.77e^{-024}$	$6.10e^{-048}$	2.97^{-095}	$7.06e^{-190}$	0
5	2.889	0.264	0.0267	0.0005	$2.60e^{-007}$	$5.91e^{-014}$	$3.06e^{-027}$	$8.27e^{-054}$	6.01^{-107}	$3.19e^{-213}$	0
6	2.657	0.185	0.0163	0.0002	$4.8e^{-008}$	$2.18e^{-015}$	$4.17e^{-030}$	$1.60e^{-059}$	2.39^{-118}	$5.30e^{-236}$	0

$$F = \begin{cases} f_1(x) = x_2x_3 \oplus x_1x_3 \oplus x_1x_2x_3 \\ f_2(x) = x_2 \\ f_3(x) = x_1 \oplus x_3 \oplus x_1x_3 \end{cases}$$

The Boolean functions are responsible for generating IGs having positive edges. The IG and the BN state space with five single ton attractors $(000 \longrightarrow 000)$,$(001 \longrightarrow 001)$,$(010 \longrightarrow 010)$, $(101 \longrightarrow 101)$ and $(111 \longrightarrow 111)$ are shown in Fig.7(a) and Fig.7(b) and truth table of the functions is shown in Table 4.

Table 4 Truth Table of three Boolean functions whose BN state space has more number of singleton attractors than p-adic attractors.

x_1	x_2	x_3	f_1	f_2	f_3
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	0	0	1
1	0	1	1	0	1
1	1	0	0	1	1
1	1	1	1	1	1

Theorem 4 *If $IG(F)$ has only negative edges, then $BN(F)$ has two length cycles or periodic attractors of length two $(p = 2)$ are more than the p-adic attractors where $p = 1,3,\ldots,2^n$.*

Proof: For 1-variable Boolean function, the set $S_1 = \{11,01,00\}$ is used for generating IG having negative edges only. The BN state space of n-variable consists of cycles length $1,2,\ldots,2^n$. In case of 1-variable the maximum length of a cycle is 2. Here the function $f_1(x) = (01)_2$ (Here most significant bit(MSB) is 0 and the least significant bit (LSB) 1) is used to generate an IG having one negative edge as the rest two functions are constant as per the definition of IG. The IG and the corresponding BN state space for the function $f_1(x)$ is shown in Fig.8(a) and Fig.8(b).

From the above *BN*s we observe that there is only one cycle having length two and no periodic cycle of length one. In this case, the Hamming weight of the least significant bit(LSB) is either same or greater as compared to the most significant bit(MSB) bit. Here, the vectors in domain have less Hamming weight from top to bottom and as per the truth table representation of Boolean functions, the *LSB* of these Boolean functions are mapped to the top vectors of the domain. On the other hand, *MSB*'s of

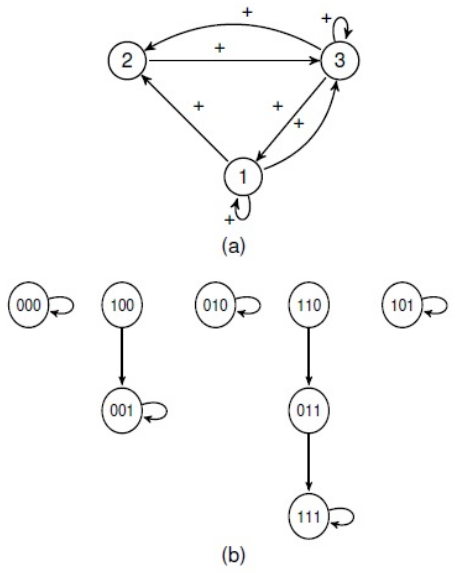

Fig. 7 (a) IG of the three Boolean functions $f_1(x)$, $f_2(x)$ and $f_3(x)$ and (b) BN state space of the three Boolean functions $f_1(x)$, $f_2(x)$ and $f_3(x)$.

Fig. 8 (a) IG of 1-variable Boolean function $f_1(x) = 1$ and (b) BN of 1-variable Boolean function $f_1(x) = 1$.

these Boolean functions are mapped to the bottom vectors of the domain. Hence, cycles having length two are generated larger as compared to other periodic cycles. The set $S_2 = \{\{0000\},\{0001\},\{0101,0011\},\{0111\},\{1111\}\}$ (The truth able format is shown in Table 5) is generated using Cartesian product of 1-variable Boolean functions. This preserve the same property, that two bit positions from *LSB* has Hamming weight either same or less with respect to the Hamming weight of two bits from *MSB*. So, the BN state space generated by this type of Boolean functions has more number of cycles having length two as shown in Fig.9(b).

Form Table 5, which contains a set of 2-variable Boolean functions defined as:

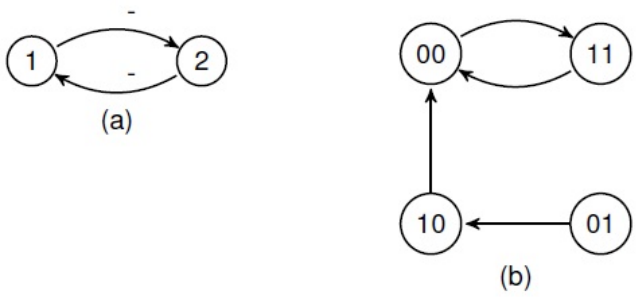

Fig. 9 (a) IG of the Boolean functions $f(x) = x_1 \oplus x_1x_2 \oplus 1$ and $g(x) = 1$ and (b) BN state space of the Boolean functions $f(x) = x_1 \oplus x_1x_2 \oplus 1$ and $g(x) = 1$.

Table 5 Shows all 2-variable Boolean functions responsible for generating BN state space having more number singleton attractor.

x_1	x_2	f_1	f_2	f_3	f_4	f_5	f_6
0	0	0	1	1	1	1	1
0	1	0	0	1	0	1	1
1	0	0	0	0	1	1	1
1	1	0	0	0	0	0	1

$$F = \begin{cases} f_1(x) = 0 \\ f_2(x) = 1 \\ f_3(x) = x_1x_2 \oplus x_2 \oplus 1 \\ f_4(x) = x_1x_2 \oplus x_1 \oplus 1 \\ f_5(x) = x_1 \oplus x_2 \oplus 1 \\ f_6(x) = x_1x_2 \oplus x_1 \oplus x_2 \oplus 1 \end{cases}$$

The above Boolean functions are used for enumerating IGs having negative edges and the corresponding BN contains more number of periodic cycles having length two. From the above six functions, taking any two functions the total number of IG or the corresponding BN state spaces are ${}^nP_k = {}^6P_2 = 30$. Here, it is worth to show that the average number of two length cycle is more than other periodic cycles and so on. The total number of one length cycle $\psi = \psi_{00} + \psi_{01} + \psi_{10} + \psi_{11}$, where ψ_{00} is the number of one length cycle yield the vertex "00". By taking any two functions, maps to the vertex "00" is zero as there is only one function having bit string 0 in *LSB*, whereas the rest functions (five functions) have bit string 1 in *LSB*. Similarly, for the vertex "11", total number of one length cycle is $\psi_{11} = 0$. For the vertex "10", total number of one length cycle is $\psi_{10} = 9/30 = 0.3$ and for the vertex "01", the total number of one length cycle is $\psi_{10} = 9/30 = 0.3$. So, the average number of one length cycle is $\psi^1_{avg} = 0 + 0.3 + 0.3 + 0 = 0.6$.

Here, every pair of vertices's generates cycle of length two. The average number of cycles having length two is $\psi^2_{avg} = 1.33556$, which is greater than the number of average cycle having length one. Boolean functions are recursively generated from $n-$variable to $(n+1)-$variable, therefore the recursive procedure using the Cartesian product also preserves the same property for the next higher variable.

The above property is also elaborated using **Markov chain with transition matrix** which is defined in (7) and (8)

Illustration:

For instance the initial transition matrix $p^1_{i,j}$ generated by the Boolean functions of 2-variable as follows.

$$p^1_{i,j} = \begin{bmatrix} 0 & 0.66667 & 0.66667 & 0.66667 \\ 0.2 & 0.3 & 0.3 & 0.2 \\ 0.2 & 0.3 & 0.3 & 0.2 \\ 0.66667 & 0.66667 & 0.66667 & 0 \end{bmatrix}$$

The average number of single length cycle or singleton attractor is $\psi^1 = \sum_{i,j=1}^{4} P_{i,j}$, $\forall\ i = j$ i.e. $\psi^1_{avg} = 0+0.3+0.3+0 = 0.6$.

Then the average number of cycle having length two is the matrix multiplication of $p^1_{i,j} \times p^1_{i,j}$ by assigning the diagonal values to zero, which is denoted by $p^2_{i,j}$, the corresponding matrix is given below.

$$p^2_{i,j} = \begin{bmatrix} 0 & 0.66667 & 0.66667 & 0.66667 \\ 0.2 & 0 & 0.3 & 0.2 \\ 0.2 & 0.3 & 0 & 0.2 \\ 0 & 0.66667 & 0.66667 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 0.66667 & 0.66667 & 0.66667 \\ 0.2 & 0 & 0.3 & 0.2 \\ 0.2 & 0.3 & 0 & 0.2 \\ 0 & 0.66667 & 0.66667 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0.51111 & 0.16111 & 0.16111 & 0.066667 \\ 0.19333 & 0.15666 & 0.06666 & 0.19333 \\ 0.19333 & 0.06666 & 0.15666 & 0.19333 \\ 0.066667 & 0.16111 & 0.16111 & 0.51111 \end{bmatrix}$$

So, the average number of cycle having length two is $\psi^2 = 0.51111 + 0.15666 + 0.15666 + 0.51111 = 1.33556$. Similarly, by using (7), the cycle length 3 and 4 are $\psi^3 = 0.266963$ and $\psi^4 = 0.0199533$ respectively. From the above $\psi^2 > \psi^1 > \psi^3 > \psi^4$. Hence, the average number of cycles having length two is more as compared to other periodic cycles. Therefore, all the Boolean functions which are generated using the recursive procedure also preserve the same property for the next higher variable. Different average cycle length up to $6-$variable is shown in Table 6. The bar plot of average number of 2-periodic attractors of these Boolean network state space are also given in Fig.10.

Table 6 Shows average number of 2-adic attractors are more than other attractors in Boolean network state space up to 6-variable.

n	ψ^1	ψ^2	ψ^3	ψ^4	ψ^5	ψ^6	ψ^7	ψ^8	ψ^9	ψ^{10}	ψ^{11}
1	0.667	0.889	-	-	-	-	-	-	-	-	-
2	0.6	1.3356	0.2669	0.0199	-	-	-	-	-	-	-
3	0.369	2.1406	0.4845	0.0535	0.0016	$1.531e^{-006}$	$1.507e^{-012}$	$1.461e^{-024}$	-	-	-
4	0.389	2.3708	0.1581	0.0069	$2.773e^{-005}$	$6.1677e^{-010}$	$3.102e^{-019}$	$7.856e^{-038}$	5.037^{-075}	$2.071e^{-149}$	$3.502e^{-298}$
5	0.246	2.291	0.066	0.0011	$8.172e^{-007}$	$5.882e^{-013}$	$3.099e^{-025}$	$8.620e^{-050}$	6.672^{-099}	$4.001e^{-197}$	0
6	0.273	2.192	0.042	0.00033	$6.750e^{-008}$	$4.061e^{-015}$	$1.527e^{-029}$	$2.161e^{-058}$	4.335^{-116}	$1.745e^{-231}$	0

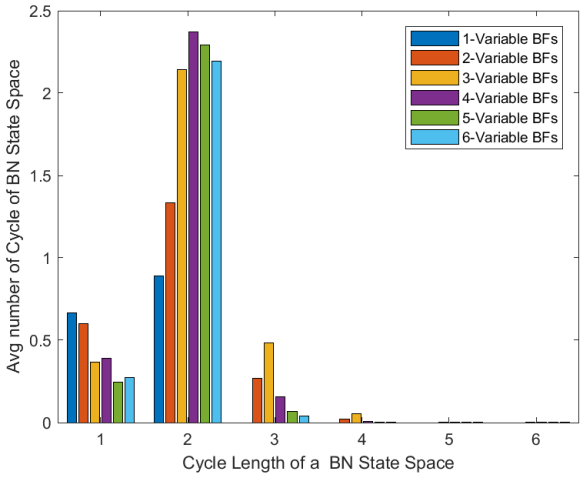

Fig. 10 Bar plot of all Boolean network state space up to 6-variable Boolean function.

Illustration: For $n = 3$, F is defined as given below.

$$F = \begin{cases} f_1(x) = x_1 \oplus x_2 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus 1 \\ f_2(x) = x_1 \oplus x_3 \oplus x_1x_2 \oplus x_1x_3 \oplus 1 \\ f_3(x) = x_2 \oplus x_3 \oplus x_1x_2 \oplus x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus 1 \end{cases}$$

The above Boolean functions are responsible for generating IGs having negative edges. The IG and the corresponding BN state space with three two periodic cycles $(000 \longrightarrow 111 \longrightarrow 000)$, $(001 \longrightarrow 011 \longrightarrow 001)$ and $(100 \longrightarrow 110 \longrightarrow 100)$ are shown in Fig.11(a) and Fig.11(b) and truth table of Boolean functions are shown in Table 7.

Table 7 Truth Table of three Boolean functions whose BN state space has more number of cycles having length two than p-adic attractors.

x_1	x_2	x_3	f_1	f_2	f_3
0	0	0	1	1	1
0	0	1	0	1	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	0	0	0

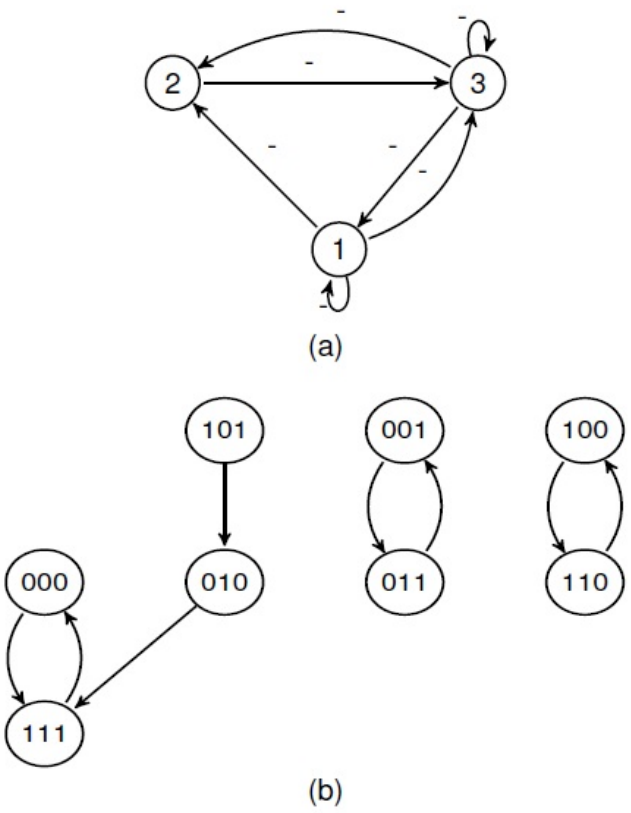

Fig. 11 (a) IG of the three Boolean function $f_1(x)$, $f_2(x)$ and $f_3(x)$ and (b) BN state space of the three Boolean function $f_1(x)$, $f_2(x)$ and $f_3(x)$.

4 A state transition model of the cell cycle network

A BN is a discrete dynamical model of gene regulatory networks/protein protein interaction networks. In this model, we assume proteins/genes are to be the nodes of the network and assign a binary value $S_i(t) \in \{0,1\}$ for each node at a time t. The overall expression level of all the genes/proteins at time t with n components is denoted by the vector $S(t) = [S_1(t), S_2(t), S_3(t), \ldots, S_n(t)]$. The state of the network is updated (for every step) according to the following rule (9)[24][11].

$$S_i(t+1) = \begin{cases} 0, & \sum_j a_{ij} S_j(t) > 0, \\ 1, & \sum_j a_{ij} S_j(t) < 0 \\ S_i(t), & \sum_j a_{ij} S_j(t) = 0 \end{cases} \tag{9}$$

where $i, j = 1,2,3,\ldots,n$ and the value of a_{ij} usually takes $-1, 1$ or 0 for interaction activation, interaction inhibition or no reaction at all, respectively. Hence $S(t)$ ranges from $[0,0,0,\ldots,0]$ (all entries of zeros) to $[1,1,1,\ldots,1]$(all entries of ones), and there are 2^n possible states. The state of all genes are updated synchronously according to the representing Boolean functions. The number of attractor depends upon the regulatory rule. A consecutive regulatory rules like $S(t), S(t+1), \ldots, S(t+p)$ is said to be a cycle of period p, if $S(t) = S(t+p)$ and $S(t) = S(t+1)$ is said to be singleton attractor. The number of singleton attractors depends upon the regulatory rule of the network. If the regulatory rules are given by $S_i(t+1) = S_i(t)$ for all i, then there are 2^n number of singleton attractors. In worst case it would take $O(2^n)$ time for identifying all singleton attractors in a network. Therefore, it is essential to develop an algorithm for identifying all singleton attractors without looking into all 2^n states. For this purpose, a recursive procedure has been proposed based on decomposition of a n-variable Boolean function into two $(n-1)$-variable Boolean function to reduce the network state space.

An example of finding the singleton attractors To verify the above procedure, we apply the cell cycle network Fission Yeast to identify singleton attractors, which is well studied in [24][11]. As shown in the Fig.12, the network consists of 10 nodes and 23 edges. Previously, it was discussed in [11], that there are 13 singleton attractors among $2^{10} = 1024$ states in the Boolean network state space of the cell cycle network as shown in the Table 8. Now, we will discuss how the decomposition of Boolean functions can reduce the search space for finding singleton attractors. All Boolean functions of Fission Yeast cell cycle represented by $f = \{f_1, f_2, \ldots, f_{10}\}$, is derived through the regulating rule discussed above. These are 10-variable Boolean functions having length 2^{10}. Let the decomposition of a Boolean function f_i into two $(n-k)$-variable Boolean function $f_i^{'}$ and $f_i^{''}$ such that $f_i = f_i{'} f_i{''}$. where $i = 1,2,\ldots,n$ and $k = i$. Here, after decomposition of the Boolean function f_1, we find two 9-variable Boolean functions $f_1{'}$ and $f_1{''}$ where $w(f_1{'}) = w(f_1{''}) = 0$. So, all the singleton attractors are in $2^9 = 512$ states of the network state space as $w(f_1{'}) = 0$. Now, similarly f_2 is decomposed into two 9-variable Boolean functions $f_2{'}$ and $f_2{''}$. Here $w(f_2{'}) > w(f_2{''})$. But $f_2{'}$ can not be used for generating singleton attractors as $w(f_1{'}) = 0$. So, $f_2{''}$ is further discomposed into two 8-variable Boolean functions $f_{21}{''}$ and $f_{22}{''}$. Here we observe that, $w(f_{21}{''}) = w(f_{22}{''}) = 0$. But $f_{21}{''}$ is not associated with singleton attractors and the state space is reduced to $2^8 = 256$ states in which singleton attractors are present. Now, we have to find out all S_i(Singleton attractors) such that $S_i(t+1) = S_i(t)$ within 256 network state space instead of 1024 network state space of the cell cycle Fission Yeast network and the corresponding singleton attractors are shown in Table 8.

5 Conclusions

The article introduces the method of generating IG using Boolean function decomposition. The proposed decomposition mechanism is found to be effective and practical as it is based on truth table instead of Boolean expression. The process of generation of $n-$variable Boolean functions starting from the three 1-variable Boolean functions

Table 8 All attractors of the dynamics of the network model of fission yeast cell cycle network.

Attractor	Basin size	Start	Sk	Cdc2/Cdc13	Ste9	Rum1	Slp1	*Cdc2/Cdc13**	Wee1/Mik1	Cdc25	PP
1	762	0	0	1	0	1	0	0	0	1	0
2	208	0	0	1	0	0	0	1	0	0	0
3	0	0	0	1	0	1	0	1	0	1	0
4	0	0	0	1	0	1	0	0	0	0	0
5	18	0	0	1	0	0	0	0	0	0	0
6	18	0	0	0	0	1	0	0	0	0	0
7	2	0	0	1	0	1	0	1	0	0	0
8	2	0	0	0	0	1	0	0	0	1	0
9	2	0	0	0	1	0	0	1	0	0	0
10	2	0	0	0	0	1	0	1	0	0	0
11	2	0	0	1	0	0	0	0	0	1	0
12	2	0	0	1	0	0	0	1	0	1	0
13	2	0	0	0	0	1	0	1	0	1	0

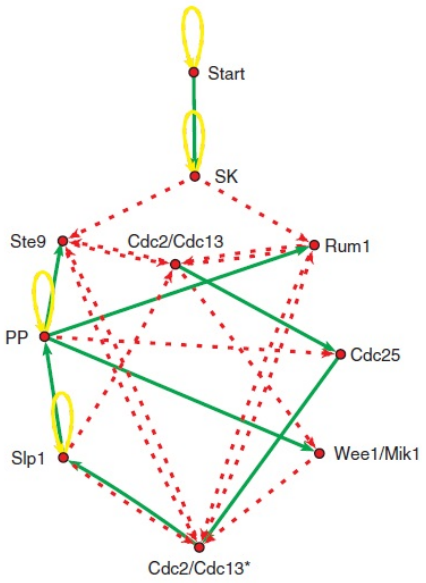

Fig. 12 The cell-cycle network of Fission Yeast. The green solid and pink dashed arrows represent positive and negative interactions, respectively[11].

is noteworthy. Then, by focusing on positive or negative edges, the presence of several attractors or the presence of cyclic attractors is seen. The average numbers of one and two periodic attractors are more in the corresponding BNs. The important point is to detect the cycles with the help of the nature of the Boolean functions instead of going through all possible 2^n states. A number of observations are incorporated for ease of understanding and clear comprehension, showing the truth table and average number of cycles having various lengths in BNs. We feel that the novel results reported in this work will be useful for the biologists in ascertaining actual biological behavior from the abstract notion of mathematical behavior. In general, large variety of properties can be statically derived from Boolean functions based on interaction graph.

Competing Interests: The authors declare no competing interests.

References

1. Akutsu, T., Kuhara, S., Maruyama, O., Miyano, S.: A system for identifying genetic networks from gene expression patterns produced by gene disruptions and overexpressions. Genome Inform **9**, 151–160 (1998)
2. Ay, F., Xu, F., Kahveci, T.: Scalable steady state analysis of boolean biological regulatory networks. PloS one **4**(12), e7992 (2009)
3. Bao, Z., Li, X., Zan, X., Shen, L., Ma, R., Liu, W.: Signalling pathway impact analysis based on the strength of interaction between genes. IET Syst Biol. **10**(4), 147–152 (2016)
4. Bornholdt, S.: Less is more in modeling large genetic networks. Science **310**(5747), 449–451 (2005)
5. Buck, V., Ng, S., Ruiz-Garcia, A.B., Papadopoulou, K., Bhatti, S., Samuel, J.M., Anderson, M., Millar, J.B., McInerny, C.J.: Fkh2p and sep1p regulate mitotic gene transcription in fission yeast. J. Cell Sci. **117**(23), 5623–5632 (2004)
6. Chai, L.E., Loh, S.K., Low, S.T., Mohamad, M.S., Deris, S., Zakaria, Z.: A review on the computational approaches for gene regulatory network construction. Comput.Biol.Med **48**, 55–65 (2014)
7. Chaos, A., Aldana, M., Espinosa-Soto, C., de León, B.G.P., Arroyo, A.G., Alvarez-Buylla, E.R.: From genes to flower patterns and evolution: dynamic models of gene regulatory networks. J Plant Growth **25**(4), 278–289 (2006)
8. Cheng, D., Qi, H.: A linear representation of dynamics of boolean networks. Trans. Autom. Control **55**(10), 2251–2258 (2010)
9. Cheng, D., Qi, H., Li, Z.: Model construction of boolean network via observed data. IEEE Trans Neural Netw **22**(4), 525–536 (2011)
10. Ching, W.K., Chen, X., Tsing, N.K.: Generating probabilistic boolean networks from a prescribed transition probability matrix. IET Syst Biol. **3**(6), 453–464 (2009)
11. Davidich, M.I., Bornholdt, S.: Boolean network model predicts cell cycle sequence of fission yeast. PloS one **3**(2), e1672 (2008)
12. Devloo, V., Hansen, P., Labbé, M.: Identification of all steady states in large networks by logical analysis. Bull Math Biol . **65**(6), 1025–1051 (2003)
13. Dougherty, E.R., Pal, R., Qian, X., Bittner, M.L., Datta, A.: Stationary and structural control in gene regulatory networks: basic concepts. Int J Syst Sci **41**(1), 5–16 (2010)
14. Espinosa-Soto, C., Padilla-Longoria, P., Alvarez-Buylla, E.R.: A gene regulatory network model for cell-fate determination during arabidopsis thaliana flower development that is robust and recovers experimental gene expression profiles. Plant Cell. **16**(11), 2923–2939 (2004)
15. Feng, W., Yang, S.X., Wu, H.: On delayed uncertain genetic regulatory networks: robust stability analysis. Int. J. Comput. Math **88**(12), 2448–2463 (2011)
16. Glass, L., Kauffman, S.A.: The logical analysis of continuous, non-linear biochemical control networks. J. Theor. Biol. **39**(1), 103–129 (1973)
17. Grinstead, C.M., Snell, J.L.: Introduction to probability. American Math. Soc. (2012)
18. Heidel, J., Maloney, J., Farrow, C., Rogers, J.: Finding cycles in synchronous boolean networks with applications to biochemical systems. Int. J. Bifurc. Chaos **13**(03), 535–552 (2003)
19. Huang, S.: Gene expression profiling, genetic networks, and cellular states: an integrating concept for tumorigenesis and drug discovery. Int J Mol Med **77**(6), 469–480 (1999)
20. Huang, S.: Cell state dynamics and tumorigenesis in boolean regulatory networks (2006)
21. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. **22**(3), 437–467 (1969)
22. Kauffman, S.A.: The origins of order: Self-organization and selection in evolution. Oxford University Press, USA (1993)
23. Lee, S., Ko, J., Tan, X., Patel, I., Balkrishnan, R., Chang, J.: Markov chain modelling analysis of hiv/aids progression: A race-based forecast in the united states. Indian J. Pharm. Sci. **76**(2), 107 (2014)
24. Li, F., Long, T., Lu, Y., Ouyang, Q., Tang, C.: The yeast cell-cycle network is robustly designed. Proceedings of the National Academy of Sciences of the United States of America **101**(14), 4781–4786 (2004)
25. Li, H., Wang, Y.: Robust stability and stabilisation of boolean networks with disturbance inputs. Int J Syst Sci. **48**(4), 750–756 (2017)

26. Mendoza, L., Xenarios, I.: A method for the generation of standardized qualitative dynamical systems of regulatory networks. Theor Biol Med Model **3**(1), 13 (2006)
27. Mochizuki, A.: An analytical study of the number of steady states in gene regulatory networks. J. Theor. Biol. **236**(3), 291–310 (2005)
28. Needham, C.J., Bradford, J.R., Bulpitt, A.J., Westhead, D.R.: A primer on learning in bayesian networks for computational biology. PLOS Comput. Biol. **3**(8), e129 (2007)
29. Novak, B., Pataki, Z., Ciliberto, A., Tyson, J.J.: Mathematical model of the cell division cycle of fission yeast. Chaos: An Inte J. of Non Sci. **11**(1), 277–286 (2001)
30. Pal, R., Ivanov, I., Datta, A., Bittner, M.L., Dougherty, E.R.: Generating boolean networks with a prescribed attractor structure. Bioinformatics **21**(21), 4021–4025 (2005)
31. Paulevé, L., Richard, A.: Static analysis of boolean networks based on interaction graphs: A survey. Electron Notes Theor Comput Sci. **284**, 93–104 (2012)
32. Rejc, Z., Magdevska, L., Trselic, T., Osolin, T., Vodopivec, R., Mraz, J., Pavliha, E., Zimic, N., Cvitanovic, T., Rozman, D., Moskon, M., Mraz, M.: Computational modeling of genome-scale metabolic networks and its application to cho cell cultures. Comput.Biol.Med (2017)
33. Remy, É., Ruet, P., Thieffry, D.: Graphic requirements for multistability and attractive cycles in a boolean dynamical framework. Adv. Appl. Math **41**(3), 335–350 (2008)
34. Rout, R.K., Choudhury, P.P., Sahoo, S., Ray, C.: Partitioning 1-variable boolean functions for various classification of n-variable boolean functions. Int J Comput Math. **92**(10), 2066–2090 (2015)
35. Rout, R.K., Pal Choudhury, P., Sahoo, S.: Classification of boolean functions where affine functions are uniformly distributed. Journal of Discrete Mathematics **2013** (2013)
36. Saadatpour, A., Albert, R., Reluga, T.C.: A reduction method for boolean network models proven to conserve attractors. SIAM J Appl Dyn Syst **12**(4), 1997–2011 (2013)
37. Sahoo, S., Choudhury, P.P., Chakraborty, M.: Characterization of any non-linear boolean function using a set of linear operators. arXiv preprint arXiv:0808.1641 (2008)
38. Seixas, F.L., Zadrozny, B., Laks, J., Conci, A., Saade, D.C.M.: A bayesian network decision model for supporting the diagnosis of dementia, alzheimer s disease and mild cognitive impairment. **51**, 140–158 (2014)
39. Shmulevich, I., Dougherty, E.R.: Probabilistic Boolean networks: the modeling and control of gene regulatory networks. SIAM (2010)
40. SIPARI, P.: Structured system models part 2. directed graphs and boolean matrices. Int J Syst Sci **22**(6), 1071–1092 (1991)
41. Thieffry, D.: Dynamical roles of biological regulatory circuits. Brief. Bioinform **8**(4), 220–225 (2007)
42. Thomas, R., d'Ari, R.: Biological feedback. CRC press (1990)
43. Trepode, N.W., de Farias, C.R., Barrera, J.: A pattern-oriented specification of gene network inference processes. Comput.Biol.Med **43**(10), 1415–1427 (2013)
44. Veliz-Cuba, A.: Reduction of boolean network models. J. Theor. Biol. **289**, 167–172 (2011)
45. Vichniac, G.Y.: Boolean derivatives on cellular automata. PHYSICA D **45**(1-3), 63–74 (1990)
46. Wu, C.H., Sahoo, D., Arvanitis, C., Bradon, N., Dill, D.L., Felsher, D.W.: Correction: Combined analysis of murine and human microarrays and chip analysis reveals genes associated with the ability of myc to maintain tumorigenesis. PLoS Gene **9**(10) (2013)
47. Xiao, M., Zheng, W.X., Cao, J.: Stability and bifurcation of genetic regulatory networks with small rnas and multiple delays. Int. J. Comput. Math **91**(5), 907–927 (2014)
48. Zhang, F., Hu, Y., Jia, Y., Xie, M.: New constructions of balanced boolean functions with high nonlinearity and optimal algebraic degree. Int. J. Comput. Math **89**(10), 1319–1331 (2012)
49. Zhou, Y., Xie, M., Xiao, G.: On cross-correlation properties of boolean functions. In: Communications and Networking in China, 2009. ChinaCOM 2009. Fourth International Conference on, pp. 1–5. IEEE (2009)
50. Rosenblueth, David A., Stalin Muñoz, Miguel Carrillo, and Eugenio Azpeitia. "Inference of Boolean networks from gene interaction graphs using a SAT solver." In International Conference on Algorithms for Computational Biology, pp. 235-246. Springer, Cham, (2014).
51. Chaves, M., and Laurent T.:"Analysis tools for interconnected Boolean networks with biological applications." Frontiers in physiology 9 (2018).
52. Chaves, M., Figueiredo, D. and Martins, M. A. : "Boolean dynamics revisited through feedback interconnections." Natural Computing (2018): 1-21.
53. Gadouleau, M., Richard, A., & Fanchon, E.: Reduction and fixed points of boolean networks and linear network coding solvability. IEEE Transactions on Information Theory, 62(5), 2504-2519(2018).
54. Li, Bo. "Graphical reduction of probabilistic boolean networks." In Control Conference (CCC), 2017 36th Chinese, pp. 1430-1434. IEEE, (2017).

A Boolean functions (BFs) are responsible to generate IGs having positive/negative edges only from 2 to 4-variable

Table A1. Shows Boolean functions (BFs) responsible to generate IGs having positive edges only from 2 to 4-variable

2-variable BFs	1100100010100000-51360	1110111011001000-61128	1111110011111100-64764
(Number of BFs =6)	1100100011000000-51392	1110111011001100-61132	1111111000000000-65024
0000-0	1100100011001000-51400	1110111011101000-61160	1111111010000000-65152
1000-8	1100110000000000-52224	1110111011101010-61162	1111111010001000-65160
1010-10	1100110010000000-52352	1110111011101100-61164	1111111010100000-65184
1100-12	1100110010001000-52360	1110111011101110-61166	1111111010101000-65192
1110-14	1100110010100000-52384	1110111011110000-61168	1111111010101010-65194
1111-15	1100110010101000-52392	1110111011111000-61176	1111111011000000-65216
3-variable BFs	1100110011000000-52416	1111000000000000-61440	1111111011001000-65224
(Number of BFs =20)	1100110011001000-52424	1111000010000000-61568	1111111011001100-65228
00000000-0	1100110011001100-52428	1111000010001000-61576	1111111011100000-65248
10000000-128	1100110011100000-52448	1111000010100000-61600	1111111011101000-65256
10001000-136	1110000000000000-57344	1111000010101000-61608	1111111011101010-65258
10100000-160	1110000010000000-57472	1111000011000000-61632	1111111011101100-65260
10101000-168	1110000010001000-57480	1111000011001000-61640	1111111011101110-65262
10101010-170	1110000010100000-57504	1111000011100000-61664	1111111011110000-65264
11000000-192	1110000011000000-57536	1111000011110000-61680	1111111011111000-65272
11001000-200	1110000011100000-57568	1111100000000000-63488	1111111011111010-65274
11001100-204	1110100000000000-59392	1111100010000000-63616	1111111011111100-65276
11100000-224	1110100010000000-59520	1111100010001000-63624	1111111011111110-65278
11101000-232	1110100010001000-59528	1111100010100000-63648	1111111100000000-65280
11101010-234	1110100010100000-59552	1111100010101000-63656	1111111110000000-65408
11101100-236	1110100010101000-59560	1111100010101010-63658	1111111110001000-65416
11101110-238	1110100011000000-59584	1111100011000000-63680	1111111110100000-65440
11110000-240	1110100011001000-59592	1111100011001000-63688	1111111110101000-65448
11111000-248	1110100011100000-59616	1111100011001100-63692	1111111110101010-65450
11111010-250	1110100011101000-59624	1111100011100000-63712	1111111111000000-65472
11111100-252	1110101000000000-59904	1111100011101000-63720	1111111111001000-65480
11111110-254	1110101010000000-60032	1111100011110000-63728	1111111111001100-65484
11111111-255	1110101010001000-60040	1111100011111000-63736	1111111111100000-65504
4-variable BFs	1110101010100000-60064	1111101000000000-64000	1111111111101000-65512
(Number of BFs =192)	1110101010101000-60072	1111101010000000-64128	1111111111101010-65514
0000000000000000-0	1110101010101010-60074	1111101010001000-64136	1111111111101100-65516
1000000000000000-32768	1110101011000000-60096	1111101010100000-64160	1111111111101110-65518
1000000010000000-32896	1110101011001000-60104	1111101010101000-64168	1111111111110000-65520
1000100000000000-34816	1110101011001100-60108	1111101010101010-64170	1111111111111000-65528
1000100010000000-34944	1110101011100000-60128	1111101011000000-64192	1111111111111010-65530
1000100010001000-34952	1110101011101000-60136	1111101011001000-64200	1111111111111100-65532
1010000000000000-40960	1110101011101010-60138	1111101011001100-64204	1111111111111110-65534
1010000010000000-41088	1110101011110000-60144	1111101011100000-64224	1111111111111111-65535
1010000010100000-41120	1110110000000000-60416	1111101011101000-64232	
1010100000000000-43008	1110110010000000-60544	1111101011101010-64234	
1010100010000000-43136	1110110010001000-60552	1111101011101100-64236	
1010100010001000-43144	1110110010100000-60576	1111101011110000-64240	
1010100010100000-43168	1110110010101000-60584	1111101011111000-64248	
1010100010101000-43176	1110110010101010-60586	1111101011111010-64250	
1010100011000000-43200	1110110011000000-60608	1111110000000000-64512	
1010101000000000-43520	1110110011001000-60616	1111110010000000-64640	
1010101010000000-43648	1110110011001100-60620	1111110010001000-64648	
1010101010001000-43656	1110110011100000-60640	1111110010100000-64672	
1010101010100000-43680	1110110011101000-60648	1111110010101000-64680	
1010101010101000-43688	1110110011101100-60652	1111110010101010-64682	
1010101010101010-43690	1110110011110000-60656	1111110011000000-64704	
1010101011000000-43712	1110111000000000-60928	1111110011001000-64712	
1010101011001000-43720	1110111010000000-61056	1111110011001100-64716	
1010101011100000-43744	1110111010001000-61064	1111110011100000-64736	
1100000000000000-49152	1110111010100000-61088	1111110011101000-64744	
1100000010000000-49280	1110111010101000-61096	1111110011101010-64746	
1100000011000000-49344	1110111010101010-61098	1111110011101100-64748	
1100100000000000-51200	1110111011000000-61120	1111110011110000-64752	
1100100010000000-51328	1110111011100000-61152	1111110011111000-64760	
1100100010001000-51336			

Table A2. Shows Boolean functions (BFs) responsible to generate IGs having negative edges only from 2 to 4-variable

2-variable BFs	0000000000010011-19	0101010101110111-21879	0011011100111111-14143
(Number of BFs =6)	0000000100010011-275	0000001101110111-887	0000111100111111-3903
0000-0	0001000100010011-4371	0001001101110111-4983	0001111100111111-7999
0001-1	0000010100010011-1299	0011001101110111-13175	0011111100111111-16191
0101-5	0000001100010011-787	0000011101110111-1911	0000000001111111-127
0011-3	0001001100010011-4883	0001011101110111-6007	0000000101111111-383
0111-7	0000000000110011-51	0101011101110111-22391	0001000101111111-4479
1111-15	0000000100110011-307	0011011101110111-14199	0000010101111111-1407
	0001000100110011-4403	0111011101110111-30583	0001010101111111-5503
3-variable BFs	0000010100110011-1331	0000111101110111-3959	0101010101111111-21887
(Number of BFs =20)	0001010100110011-5427	0001111101110111-8055	0000001101111111-895
00000000-0	0000001100110011-819	0000000000001111-15	0001001101111111-4991
00000001-1	0001001100110011-4915	0000000100001111-271	0011001101111111-13183
00010001-17	0011001100110011-13107	0001000100001111-4367	0000011101111111-1919
00000101-5	0000011100110011-1843	0000010100001111-1295	0001011101111111-6015
00010101-21	0000000000000111-7	0001010100001111-5391	0101011101111111-22399
01010101-85	0000000100000111-263	0000001100001111-783	0011011101111111-14207
00000011-3	0001000100000111-4359	0001001100001111-4879	0111011101111111-30591
00010011-19	0000010100000111-1287	0000011100001111-1807	0000111101111111-3967
00110011-51	0000001100000111-775	0000111100001111-3855	0001111101111111-8063
00000111-7	0000011100000111-1799	0000000000011111-31	0101111101111111-24447
00010111-23	0000000000010111-23	0000000100011111-287	0011111101111111-16255
01010111-87	0000000100010111-279	0001000100011111-4383	0111111101111111-32639
00110111-55	0001000100010111-4375	0000010100011111-1311	0000000011111111-255
01110111-119	0000010100010111-1303	0001010100011111-5407	0000000111111111-511
00001111-15	0001010100010111-5399	0101010100011111-21791	0001000111111111-4607
00011111-31	0000001100010111-791	0000001100011111-799	0000010111111111-1535
01011111-95	0001001100010111-4887	0001001100011111-4895	0001010111111111-5631
00111111-63	0000011100010111-1815	0011001100011111-13087	0101010111111111-22015
01111111-127	0001011100010111-5911	0000011100011111-1823	0000001111111111-1023
11111111-255	0000000001010111-87	0001011100011111-5919	0001001111111111-5119
	0000000101010111-343	0000111100011111-3871	0011001111111111-13311
4-variable BFs	0001000101010111-4439	0001111100011111-7967	0000011111111111-2047
(Number of BFs =192)	0000010101010111-1367	0000000001011111-95	0001011111111111-6143
0000000000000000-0	0001010101010111-5463	0000000101011111-351	0101011111111111-22527
0000000000000001-1	0101010101010111-21847	0001000101011111-4447	0011011111111111-14335
0000000100000001-257	0000001101010111-855	0000010101011111-1375	0111011111111111-30719
0000000000010001-17	0001001101010111-4951	0001010101011111-5471	0000111111111111-4095
0000000100010001-273	0011001101010111-13143	0101010101011111-21855	0001111111111111-8191
0001000100010001-4369	0000011101010111-1879	0000001101011111-863	0101111111111111-24575
0000000000000101-5	0001011101010111-5975	0001001101011111-4959	0011111111111111-16383
0000000100000101-261	0101011101010111-22359	0011001101011111-13151	0111111111111111-32767
0000010100000101-1285	0000111101010111-3927	0000011101011111-1887	1111111111111111-65535
0000000000010101-21	0000000000110111-55	0001011101011111-5983	
0000000100010101-277	0000000100110111-311	0101011101011111-22367	
0001000100010101-4373	0001000100110111-4407	0011011101011111-14175	
0000010100010101-1301	0000010100110111-1335	0000111101011111-3935	
0001010100010101-5397	0001010100110111-5431	0001111101011111-8031	
0000001100010101-789	0101010100110111-21815	0101111101011111-24415	
0000000001010101-85	0000001100110111-823	0000000000111111-63	
0000000101010101-341	0001001100110111-4919	0000000100111111-319	
0001000101010101-4437	0011001100110111-13111	0001000100111111-4415	
0000010101010101-1365	0000011100110111-1847	0000010100111111-1343	
0001010101010101-5461	0001011100110111-5943	0001010100111111-5439	
0101010101010101-21845	0011011100110111-14135	0101010100111111-21823	
0000001101010101-853	0000111100110111-3895	0000001100111111-831	
0001001101010101-4949	0000000001110111-119	0001001100111111-4927	
0000011101010101-1877	0000000101110111-375	0011001100111111-13119	
0000000000000011-3	0001000101110111-4471	0000011100111111-1855	
0000000100000011-259	0000010101110111-1399	0001011100111111-5951	
0000001100000011-771	0001010101110111-5495	0101011100111111-22335	