Stochastic Multiple-Valued Gene Networks

Peican Zhu and Jie Han

***Abstract*—Among various approaches to modeling gene regulatory networks (GRNs), Boolean networks (BNs) and its probabilistic extension, probabilistic Boolean networks (PBNs), have been studied to gain insights into the dynamics of GRNs. To further exploit the simplicity of logical models, a multiple-valued network employs gene states that are not limited to binary values, thus providing a finer granularity in the modeling of GRNs. In this paper, stochastic multiple-valued networks (SMNs) are proposed for modeling the effects of noise and gene perturbation in a GRN. An SMN enables an accurate and efficient simulation of a probabilistic multiple-valued network (as an extension of a PBN). In a *k*-level SMN of *n* genes, it requires a complexity of $O(nLk^n)$ to compute the state transition matrix, where L is a factor related to the minimum sequence length in the SMN for achieving a desired accuracy. The use of randomly permuted stochastic sequences further increases computational efficiency and allows for a tunable tradeoff between accuracy and efficiency. The analysis of a p53-Mdm2 network and a WNT5A network shows that the proposed SMN approach is efficient in evaluating the network dynamics and steady state distribution of gene networks under random gene perturbation.**

***Index Terms*—Gene perturbation, multiple-valued logic, stochastic computation, steady state analysis, Boolean networks.**

I. Introduction

In a cell, biological functions are implemented through the interactions among genes, proteins and other molecules. However, gene networks are noisy due to the effect of stochastic fluctuations in genetic interactions [1]. Various methodologies have been proposed to model the interactions among genes [2]. As a classic logical model, Boolean networks (BNs) provide a qualitative analysis of the network dynamics [3 - 5]. Probabilistic Boolean networks (PBNs) further consider noise in a BN model [6 - 8]. Recently, stochastic Boolean networks (SBNs) have been used to efficiently implement the function of PBNs [9]. By a stochastic simulation of a PBN, an SBN trades off accuracy for efficiency and thus provides an alternative and efficient means to help understanding the dynamics of gene regulatory networks (GRNs), such as those in the oscillatory behavior of a p53-Mdm2 network [10] and the dynamic attractors in a T cell immune response network [11].

The Boolean simplification, however, may incur an accuracy loss in the modeling of complex biological networks such as a random Boolean network [12, 13]. To address this, an approach using multiple-valued variables introduces an increased level of granularity and can thus be more accurate in the modeling of a gene regulatory network (GRN) [14-17]. For examples, three states of the protein p53 is considered in [18, 19] and multiple-valued gene nodes are analyzed in a T-helper network [17]. Moreover, deterministic multiple-valued networks are analyzed in [20]. A multiple-valued analysis provides a tradeoff between the simplicity of Boolean networks and the complexity of differential equation based approaches [15]. Multiple-valued networks have also been studied in chemical reactions [21] and cognitive sciences [22].

When gene expressions are discretized into multiple values, they are considered to be not only affected by the presence of activating or repressing proteins, but also by the absence of a protein [23]. Random and probabilistic multiple-valued networks (PMNs) have respectively been studied in [16] and [24], for providing insights into the long run behavior of a network with noise. For a *k*-valued network of n genes with N network functions, however, a $k^n \times k^n$ matrix is required for an accurate analysis of the steady state distribution (SSD), resulting in a complexity of $O(nNk^{2n})$ by a PMN analysis in the computation of the state transition matrix (STM). This also requires a memory usage in the order of at least $O(k^{2n})$. Since the size of an STM (and the required memory) increase exponentially with the number of genes, the analysis of a network with a higher quantization level presents even a greater challenge. This prevents the use of an accurate analysis in the evaluation of large networks. For a network with an increased number of genes, a Markov chain Monte Carlo (MCMC) method is often used to estimate the SSD of a PBN [25] and its multiple-valued extension, PMNs [24]. An MCMC simulation is considered to produce an accurate result when a sufficient number of simulations are performed to produce a stable output; however, this number is usually required to be very large, due to the slow convergence of the MCMC method [26], thus incurring a very long simulation time.

In this paper, stochastic multiple-valued networks (SMNs) are proposed for an efficient implementation of probabilistic multiple-valued networks (PMNs), where the quantization level of a gene's state is not limited to binary. As in stochastic computation, SMNs employ random streams of multiple values to represent probabilities and computation is performed by stochastic logic. Due to stochastic fluctuations, however, the computational results obtained by an SMN are not deterministic, but probabilistic. In an SBN, it has been shown that the use of non-Bernoulli sequences of random permutations of fixed numbers of 1's and 0's as initial inputs reduces the stochastic fluctuation and produces more accurate results than using Bernoulli sequences [9]. In a *k*-valued SMN, similarly, randomly permuted sequences of fixed numbers of the k values are used to reduce the required computational complexity. It is shown in simulation results that the use of randomly permuted sequences increases the computational

This paper is submitted on April 28th, 2013. This work was supported in part by an NSERC Discovery Grant.

The authors are with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada T6G 2V4. (e-mails: {peican, jhan8}@ualberta.ca).

efficiency and allows for a tunable tradeoff between accuracy and efficiency. The required complexity for computing the STM of a k-valued network is reduced from $O(nNk^{2n})$ to $O(nLk^n)$, where L, determined by the minimum length of stochastic sequences for achieving a desired accuracy, increases slower than N.

Nevertheless, the analysis of the steady state distribution (SSD) is challenging due to the size of the STM required. However, the SSD analysis of a PMN resembles that of a finite state machine (FSM), due to their common underlying Markovian nature. An FSM is often implemented by a sequential circuit, which can be unrolled into a series of identical combinational modules by a so-called time-frame expansion in the spatial domain. A time-frame expansion of an SMN, hence, is used for an SSD analysis, which makes the SMN approach very efficient in the analysis of complex GRNs. Simulation results show that the proposed SMN approach produces very accurate results for small networks compared to a theoretical analysis. For large networks, the SMN approach using the time-frame expansion technique is more efficient than a simulation-based MCMC method. It is shown that the SMN approach reveals the oscillatory dynamics of a multiple-valued p53-Mdm2 network [19] with random gene perturbation, and that it accurately and efficiently predicts the SSD of a ternary WNT5A network [24] with gene perturbation.

The rest of the paper is organized as follows. Section II reviews the definitions of PMNs. Section III presents stochastic multiple-valued networks (SMNs) without and with gene perturbation for STM and SSD analysis. In Section IV, a multiple-valued p53-Mdm2 and a ten-gene WNT5A network are analyzed using the proposed SMN approach. Finally, Section V concludes the paper.

II. Probabilistic Multiple-valued Networks

A multiple-valued network of n genes is defined by $G(V,F)$, with a node set $V=\{x_1,x_2,\cdots,x_n\}$ and a list of sets of predictor functions $F=\{F_1,F_2,\cdots,F_n\}$ [6]. If the state of gene i is quantized into k levels, then $x_i\in\{0,\cdots,k-1\}$ for $i\in\{1,\cdots,n\}$. For $k=2$, a network is referred to as a probabilistic Boolean network (PBN), where V is a set of binary-valued nodes; for $k=3$, it is considered as a ternary network [24]. At time t, the state of a network can be described by a vector, $\boldsymbol{x}(t)=(x_1(t),x_2(t),\cdots,x_n(t))$, where the state of a gene is given by $x_i(t)\in\{0,\cdots,k-1\}$ for $i\in\{0,1,\cdots,n\}$. A network state is also referred to as a gene activity profile (GAP). For a k-valued network of n genes, hence, there are a total of k^n network states or GAPs. A GAP is also given as a decimal index. For a ternary network of n genes, a GAP is indexed by:

$$d=\sum_{i=1}^{n}x_i(t)\cdot 3^{i-1}+1, \tag{1}$$

where x_i is the state of the ith gene, $i\in\{0,1,\cdots,n\}$.

For gene i ($i\in\{1,\cdots,n\}$), the set of predictor functions is given by $F_i=\{f_1^{(i)},f_2^{(i)},\cdots,f_{l(i)}^{(i)}\}$, with each predictor function $f_{j(i)}^{(i)}:\{0,1,\cdots,k-1\}^n\to\{0,1,\cdots,k-1\}$, where $l(i)$ is the number of possible predictor functions for gene i and $l(i)$ is usually a small number [27, 28]. Due to the stochastic behavior, the next state of gene i is determined by all of its predictor functions in F_i, i.e., $f_1^{(i)},f_2^{(i)}\ldots,f_{l(i)}^{(i)}$ with probabilities $c_1^{(i)},c_2^{(i)},\cdots,c_{l(i)}^{(i)}$.

If the predictor functions are independent, there are $N=\prod_{i=1}^{n}l(i)$ possible realizations of the network, each of which is referred to as a context. Assume that the j th context is represented as $\boldsymbol{f}^j=(f_{j(1)}^{(1)},f_{j(2)}^{(2)},\cdots,f_{j(n)}^{(n)})$, where each $f_{j(i)}^{(i)}:\{0,1,\cdots,k-1\}^n\to\{0,1,\cdots,k-1\}$, for $1\le j(i)\le l(i)$, is a predictor function of gene i; the next state of a gene is determined by both the present state and the selected context.

A multiple-valued network can be modeled by a Markov chain [24], so the next state of gene i, x_i ($x_i\in\{0,1,\cdots,k-1\}$ in a k-valued network) is given by:

$$x_i^{(t+1)}=\begin{cases}0 & with\ C_i^0(\boldsymbol{S}^{(t)})=Pr(x_i^{(t+1)}=0|\boldsymbol{S}^{(t)})\\ 1 & with\ C_i^1(\boldsymbol{S}^{(t)})=Pr(x_i^{(t+1)}=1|\boldsymbol{S}^{(t)})\\ & \vdots\\ k-1 & with\ C_i^{k-1}(\boldsymbol{S}^{(t)})=Pr(x_i^{(t+1)}=k-1|\boldsymbol{S}^{(t)})\end{cases}. \tag{2}$$

where $C_i^0(\boldsymbol{S}^{(t)})+C_i^1(\boldsymbol{S}^{(t)})+\cdots+C_i^{k-1}(\boldsymbol{S}^{(t)})=1$. Thus, the transition probability from the network state (or GAP) $\boldsymbol{S}^{(t)}$ at time t to $\boldsymbol{S}^{(t+1)}$ at $t+1$ is given by:

$$Pr(\boldsymbol{S}^{(t)}\to\boldsymbol{S}^{(t+1)})=\prod_{i=1}^{n}C_i^{x_i^{(t+1)}}. \tag{3}$$

Using the decimal indices of GAPs by (1), the state transition of a ternary network is described by the state transition matrix (STM) as follows:

$$\boldsymbol{A}=\begin{bmatrix}Pr(1|1) & Pr(2|1) & \cdots & \cdots & Pr(3^n|1)\\ Pr(1|2) & Pr(2|2) & \cdots & \cdots & Pr(3^n|2)\\ \cdots & \cdots & \cdots & \cdots & \cdots\\ \cdots & \cdots & \cdots & \cdots & \cdots\\ Pr(1|3^n) & Pr(2|3^n) & \cdots & \cdots & Pr(3^n|3^n)\end{bmatrix}. \tag{4}$$

In $\boldsymbol{A}$, each entry indicates the conditional probability that the network transitions from a present state into a next state. For N realizations of the network, $\boldsymbol{A}$ can be obtained as $\boldsymbol{A}=\sum_{j=1}^{N}P_j\boldsymbol{A}_j$, where P_j ($P_j=\prod_{i=1}^{n}c_{j(i)}^{(i)}$) is the probability that the jth realization of the network emerges and $\boldsymbol{A}_j$ is the STM resulting from the jth realization [6]. Hence, the STM can be derived for a multiple-valued network with a complexity of $O(nNk^{2n})$, where N is the number of possible realizations of the network and k is the quantization level of the gene states.

External stimuli cause random gene perturbations that make the dynamics of a network an ergodic Markov chain [7]. In an ergodic Markov chain, all states are communicated and thus a steady state distribution (SSD) exists in a network. Since a perturbed gene has $k-1$ possible states, there are $(k-1)^{n_0}$ states for n_0 perturbed genes ($n_0\in\{1,\cdots,n\}$); hence, each of the perturbed states in $\boldsymbol{S}^{(t+1)}$ is selected with a probability of $[1/(k-1)]^{n_0}$. The event that no gene is perturbed, occurs with a probability of $(1-p)^n$. Hence, $\boldsymbol{S}^{(t+1)}$ is determined by the selected context if no perturbation exists, i.e. $Pr\{\boldsymbol{S}^{(t)}\to\boldsymbol{S}^{(t+1)}\}=\prod_{l=1}^{n}C_l^{x_l^{(t+1)}}$. If n_0 genes are perturbed, $\boldsymbol{S}^{(t)}\to\boldsymbol{S}^{(t+1)}$ occurs with probability $p^{n_0}\cdot(1-p)^{n-n_0}\cdot[1/(k-1)]^{n_0}$. Following [24], therefore, the state transition

probability from $\boldsymbol{S}^{(t)}$ to $\boldsymbol{S}^{(t+1)}$ in a perturbed k-valued network is given by:

$$Pr\{\boldsymbol{S}^{(t)} \to \boldsymbol{S}^{(t+1)}\} = \left(\prod_{i=1}^{n} C_i^{x_i^{(t+1)}}\right) \cdot (1-p)^n + p^{n_0} \cdot (1-p)^{n-n_0} \cdot p_0^{n_0} \cdot 1[\boldsymbol{S}^{(t)} \neq \boldsymbol{S}^{(t+1)}], \quad (5)$$

with

$$n_0 = \sum_{i=1}^{n} 1(x_i^{(t)} \neq x_i^{(t+1)}), \quad (6)$$

$$p_0 = 1/(k-1), \quad (7)$$

where p is the perturbation rate, n_0 is the number of perturbed genes, p_0 is the probability that a gene will change to a new state if perturbed, and $1(\cdot)$ is an indicator function; $1[\boldsymbol{S}^{(t)} \neq \boldsymbol{S}^{(t+1)}] = 1$ if $\boldsymbol{S}^{(t)} \neq \boldsymbol{S}^{(t+1)}$ and $1[\boldsymbol{S}^{(t)} \neq \boldsymbol{S}^{(t+1)}] = 0$ otherwise. Using (5), a perturbed state transition matrix (STM) or perturbation matrix [9, 27] can be obtained for further analysis of the steady state distribution (SSD).

III. Stochastic Multiple-valued Networks

A. Stochastic Computation for Multiple-valued Logic

In stochastic computation, probabilities are encoded into random binary bit streams. Information is carried in the statistics of the binary streams and processed by stochastic logic [29]. Usually, a probability is represented by a proportional number of bits, e.g. the mean number of 1's in a bit sequence. In Boolean logic, for example, an inverter computes the complement of a probability while the multiplication of probabilities is implemented by an AND gate with independent inputs. Thus, stochastic computation performs a probabilistic analysis in the real domain. Due to inevitable stochastic fluctuations, the computational result by stochastic logic is not deterministic but probabilistic. However, stochastic fluctuations can be reduced through the use of non-Bernoulli sequences of random permutations of fixed numbers of 1's and 0's as initial inputs. This produces more accurate results than using Bernoulli sequences [30]. Signal correlations are efficiently handled in a stochastic network by the bit-wise dependencies encoded in the random binary streams, thus making it an efficient approach to computing probabilities [30].

Stochastic computation is also applicable to the probabilistic analysis of multiple-valued signals. For a k-valued signal, the probability of each value is given in a vector $P = [p_{k-1}, p_{k-2}, \cdots, p_1, p_0]$, with $\sum_{i=0}^{k-1} p_i = 1$. This probability vector can be encoded into a multiple-valued stochastic sequence. An example is shown in Fig. 1 for a ternary signal.

"0121012102" for $\begin{cases} P(0) = 0.3 \\ P(1) = 0.4 \\ P(2) = 0.3 \end{cases}$

Fig. 1. The stochastic encoding of a ternary signal using a sequence of 10 values.

Multiple-valued logic includes the buffer, inverter, MIN (minimum), MAX (maximum) and rotator; some are defined as follows [20]:

(1) A multiple-valued buffer:

$$\mathrm{BUF}(A) = A,$$

(2) A multiple-valued inverter:

$$\mathrm{INV}(A) = (k-1) - A,$$

(3) A multiple-valued rotator $\emptyset$:

$$\emptyset(A) = \begin{cases} A+1 & A \neq k-1 \\ 0 & A = k-1 \end{cases}.$$

The following new logic operators are further defined:

(4) A multiple-valued equal or larger (EL) operator:

$$EL(A \geq a) = MAX(A, a),$$

(5) A multiple-valued equal or smaller (ES) operator:

$$ES(A \leq a) = MIN(A, a).$$

Several ternary stochastic processing elements are shown in Fig. 2, including a buffer, an inverter, an EL operator, an ES operator, a MIN, a MAX, a rotate gate and a 4-to-1 multiplexer.

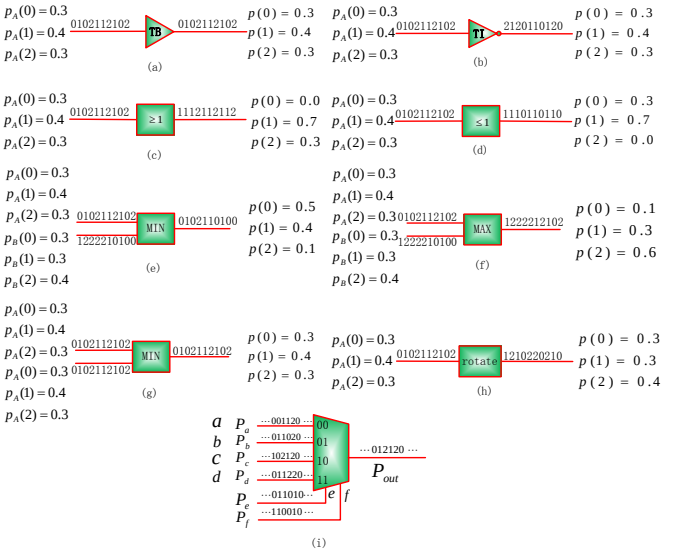

Fig. 2. Stochastic logic: (a) a ternary buffer (TB); (b) a ternary inverter (TI); (c) an EL operator; (d) an ES operator; (e) a ternary MIN with independent inputs; (f) a ternary MAX with independent inputs; (g) a ternary MIN with totally dependent inputs; (h) a ternary rotate gate; (i) a 4-to-1 multiplexer. A probabilistic computation is performed through stochastic logic operations by encoding signal probabilities into random sequences.

For the ternary MIN logic, if the two inputs are independent with probabilities A = [0.3 0.4 0.3] and B = [0.5 0.4 0.1], the output probabilities are expected to be $p(2) = p_A(2) \cdot p_B(2) = 0.3 \times 0.1 = 0.03$, $p(0) = p_A(0) + p_B(0) - p_A(0) \cdot p_B(0) = 0.65$ and $p(1) = 1 - p(0) - p(2) = 0.32$. This function can be implemented by the ternary MIN gate, as shown in Fig. 2(e), using stochastic sequences. For a sequence length of 10,000 values, the output sequence is expected to have approximately 6500 0's, 3200 1's and 300 2's. For the ternary rotate logic, if the input's signal probability is given by A = [0.3 0.4 0.3], the output's signal probability is expected to be $p(0) = p_A(2) = 0.3$, $p(1) = p_A(0) = 0.3$ and $p(2) = p_A(1) = 0.4$. This function can be implemented by the ternary rotate gate with the use of stochastic sequences (Fig. 2(h)).

For the 4-to-1 multiplexer logic in Fig. 2(i), its output is determined by its binary control signals $'ef'$. It takes the value of input a for $ef = 00$, b for $ef = 01$, c for $ef = 10$, or d for $ef = 11$. Similarly, a stochastic multiplexer takes one of the inputs as its output according to the distributions of control bits

(i.e., 00, 01, 10 and 11). Thus, the selection probabilities are encoded in the random sequences of the control bits. However, these numbers are not deterministic but probabilistic, due to inherent stochastic fluctuations. For stochastic Boolean networks, it has been shown that, compared to the use of Bernoulli sequences of independently generated binary bits (such as in a coin-flipping experiment), the effect of the fluctuation can be significantly reduced through the use of non-Bernoulli sequences of random permutations of fixed numbers of 1's and 0's for initial input probabilities [9]. In this paper, stochastic sequences of random permutations of fixed numbers of the multiple values, hereafter referred to as randomly permuted sequences, are used for encoding initial input probabilities. The use of randomly permuted sequences reduces the amount of stochastic fluctuations in a network. It will be shown in the Results and Discussion Section that the effect of fluctuation is negligible when a reasonable sequence length is used in the simulation.

B. *Stochastic Multiple-valued Networks without Perturbation*

A stochastic Boolean network (SBN) has been proposed for an instantaneous probabilistic Boolean network (PBN) [9]. In the general case that multiple quantization levels are considered, a stochastic multiple-valued network (SMN) can be constructed to model a multiple-valued gene network. As discussed previously, the next state of a gene is determined by the present state of its input genes and a set of predictor functions according to their occurring probabilities. In an SMN, these probabilities are represented by randomly permuted multiple-valued sequences and the selection of the predictor functions is implemented by a multiple-input multiplexer with properly generated control sequences. A structure of the SMN for a single gene is shown in Fig. 3.

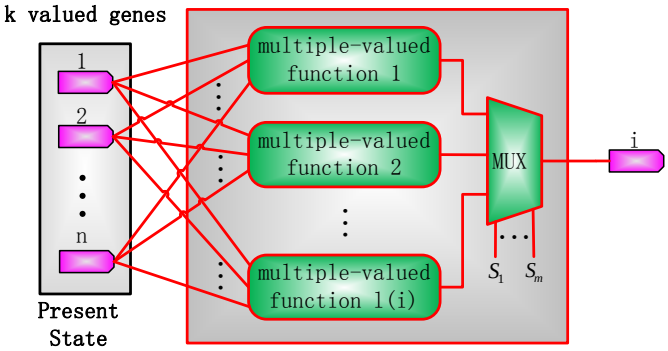

Fig. 3. A stochastic multiple-valued network (SMN) without perturbation (for a single gene i). The control sequences S_1~S_m of the multiplexer (MUX) probabilistically determine the selection of the multiple-valued functions.

If the next state of gene i is determined by $l(i)$ predictor functions, the number of control bits of the multiplexer is given by $\lceil log_2(l(i)) \rceil$. Usually, a function has only a few input variables and the number of possible predictor functions is generally small [27, 28]. By a multiplexer with control bits S_1~S_m, a function is selected in the jth BN for gene i with probability $c_{j(i)}^{(i)}$. Assume that a network transfers from state $\boldsymbol{S}^{(t)}$ to $\boldsymbol{S}^{(t+1)}$ in a context (or network function), then the transition probability for $\boldsymbol{S}^{(t)} \to \boldsymbol{S}^{(t+1)}$ is given by the probability of selecting this context. This indicates that when all the genes are considered, the SMN model in Fig. 3 accurately implements the function of (3).

C. *Stochastic Multiple-valued Networks with Perturbation*

Under external stimuli, a gene's state can be perturbed by a small chance during a transition [7]. In a k-valued network of n genes, a perturbation flag vector $\boldsymbol{\gamma}$ is used to indicate whether a gene is to be perturbed. Assume that the network goes from state $\boldsymbol{S}^{(t)}$ to $\boldsymbol{S}^{(t+1)}$ under perturbation. If each gene is to be perturbed with a probability p, the probability that the next state is totally determined by a network function (i.e., no perturbation occurs) is $(1-p)^n$. When a perturbation occurs, the state of the perturbed gene transitions to a different state: this new state is determined by the present state and the value in the perturbation flag vector $\boldsymbol{\gamma}$. Without the loss of generality, a set of transition rules can be determined, as shown in Table 1 for a ternary network. The set of rules in Table 1 can be implemented by sum and modulo operations; for $\boldsymbol{S}^{(t)} = (0,0,0,1,1,1,2,2,2)$ and $\boldsymbol{\gamma} = (0,1,2,0,1,2,0,1,2)$, as an example, the next state is given by $\boldsymbol{S}^{(t+1)} = modulo\left(\left(\boldsymbol{S}^{(t)} + \boldsymbol{\gamma}\right), 3\right) = (0,1,2,1,2,0,2,0,1)$. Hence, the perturbation in a ternary network can be implemented by the sum and modulo operations. For a network of higher levels, similar operations can be implemented for the perturbation (although not discussed in detail), while for a Boolean network, this operation is simplified to an XOR gate.

TABLE 1 STATE TRANSITION RULES FOR A GENE IN A TERNARY NETWORK UNDER PERTURBATION

Current State (x)	perturbation (γ)	Next State ($x' = modulo(x+\gamma, 3)$)
0	1	1
	2	2
1	1	2
	2	0
2	1	0
	2	1

For an SMN, therefore, if $\boldsymbol{S}^{(t)} = (x_1, x_2, \cdots, x_n)$ is the GAP or state of the network at time t; the next state $\boldsymbol{S}^{(t+1)}$ is given by:

$$\boldsymbol{S}^{(t+1)} = \begin{cases} modulo(sum(\boldsymbol{S}^{(t)}, \boldsymbol{\gamma}), k) & with\ 1-(1-p)^n, \\ \boldsymbol{f}_j(x_1, x_2, \cdots, x_n) & with\ (1-p)^n, \end{cases} \quad (8)$$

where p is the perturbation rate for each gene and $\boldsymbol{f}_j(\cdot)$ is the jth realization of the network at time t. (8) indicates that no perturbation occurs, i.e., $\gamma_i = 0$ for any $i \in \{1, \cdots, n\}$, with a probability of $(1-p)^n$. In this case, the next state $\boldsymbol{S}^{(t+1)}$ is determined by the selected context (or network function). If gene i is perturbed, γ_i in $\boldsymbol{\gamma}$ is assigned to be m $(m \neq 0)$ with a probability of $1/(k-1)$; the gene's state x_i is then changed from j to m $(m \neq j)$ with a probability of $1/(k-1)$ [24]. This state transition under perturbation is then implemented by the function of $modulo(sum(\boldsymbol{S}^{(t)}, \boldsymbol{\gamma}), k)$. In a network of n

TABLE 2 MINIMUM SEQUENCE LENGTH AND AVERAGE RUN TIME REQUIRED IN COMPUTING THE STATE TRANSITION MATRIX OF TERNARY SMNS, COMPARED TO THOSE OBTAINED BY A MARKOV CHAIN ANALYSIS [24]. n: THE NUMBER OF GENES; N: POSSIBLE NUMBER OF NETWORKS; PERTURBATION RATE $p = 0.1$; L: REQUIRED MINIMUM SEQUENCE LENGTH. ACCURACY OF THE SMN APPROACH IS MEASURED BY NORM 2 BETWEEN THE STMS OBTAINED BY THE MARKOV CHAIN ANALYSIS AND THE SMN APPROACH. AN EQUAL NUMBER OF PREDICTOR FUNCTIONS ARE RANDOMLY GENERATED FOR EACH GENE.

n	N	Number of states	SMN (Norm 2 = 0.04)			SMN (Norm 2 = 0.02)			Markov chain analysis [24]	
			L	Average run time (s)	Standard deviation	L	Average run time (s)	Standard deviation	Average run time (s)	Standard deviation
2	4	9	260	0.00276	0.00044	1,000	0.00666	0.00097	0.00344	0.00019
3	8	27	900	0.02321	0.00131	3,600	0.08361	0.00419	0.02354	0.00168
4	16	81	1,600	0.14861	0.00506	6,000	0.53357	0.00416	0.19815	0.00644
5	32	243	2,700	0.93284	0.03965	10,000	3.37799	0.01376	1.60821	0.01220
6	64	729	4,200	4.67169	0.02427	17,000	15.0681	0.03060	12.3910	0.36403
7	128	2187	6,000	24.2737	0.16698	24,000	74.7821	0.62384	119.080	5.91620
8	256	6561	10,000	136.930	3.02933	34,000	438.999	11.9537	1003.70	37.1771

TABLE 3 AVERAGE RUN TIME IN COMPUTING THE STEADY STATE DISTRIBUTION (SSD) OF SMNS, COMPARED TO THE USE OF A MARKOV CHAIN ANALYSIS [24]. ACCURACY OF THE SMN APPROACH IS MEASURED BY NORM 2 BETWEEN THE SSDS, I.E. $\|\Delta \boldsymbol{SSD}\|_2$, OBTAINED BY THE MARKOV CHAIN ANALYSIS [24] AND THE SMN APPROACH. THE STEADY STATE IS CONSIDERED TO HAVE BEEN REACHED IN 30 ITERATIONS. n: THE NUMBER OF GENES; N: POSSIBLE NUMBER OF NETWORKS; K: THE DISCRETIZATION LEVEL OF A GENE NETWORK (ALL GENES ARE ASSUMED TO HAVE THE SAME DISCRETIZATION LEVEL); PERTURBATION RATE $p = 0.1$; L: SEQUENCE LENGTH USED IN THE SIMULATION.

n	N	k	Time frame expanded SMN approach									Markov [24] Average time (s)
			L	Average time (s)	$\|\Delta \boldsymbol{SSD}\|_2$	L	Average time (s)	$\|\Delta \boldsymbol{SSD}\|_2$	L	Average time (s)	$\|\Delta \boldsymbol{SSD}\|_2$	
2	4	3	1k	0.017933	0.0255	10k	0.191410	0.0096	100k	1.385850	0.0058	0.016257
		4		0.017418	0.0282		0.178193	0.0105		1.374220	0.0072	0.020486
		5		0.020142	0.0306		0.161153	0.0125		1.633088	0.0091	0.016485
		6		0.019914	0.0312		0.175710	0.0130		1.539379	0.0106	0.049608
3	8	3	1k	0.033131	0.0283	10k	0.266318	0.0099	100k	2.616372	0.0049	0.043657
		4		0.029720	0.0303		0.274549	0.0104		2.494515	0.0054	0.093971
		5		0.028082	0.0310		0.277415	0.0116		2.540246	0.0064	0.383156
		6		0.029365	0.0323		0.274347	0.0121		2.456368	0.0073	1.050769
4	16	3	1k	0.033118	0.0293	10k	0.367657	0.0098	100k	3.207161	0.0045	0.243478
		4		0.039506	0.0302		0.357331	0.0100		2.815740	0.0046	1.629470
		5		0.033572	0.0312		0.306706	0.0107		2.844093	0.0054	8.522092
		6		0.032634	0.0317		0.309268	0.0110		2.857933	0.0059	35.32554

TABLE 4 REQUIRED MEMORY USAGE IN COMPUTING THE STEADY STATE DISTRIBUTION (SSD) OF MULTIPLE-VALUED NETWORKS BY THE MARKOV CHAIN ANALYSIS [24] AND TIME FRAME EXPANDED SMN APPROACH, GIVEN BY Mem_{MCA} AND Mem_{SMN} RESPECTIVELY. 50 ITERATIONS ARE PERFORMED IN EACH SIMULATION. n: THE NUMBER OF GENES; K: THE DISCRETIZATION LEVEL OF A GENE NETWORK; PERTURBATION RATE $p = 0.1$; SEQUENCE LENGTH: L = 30K. TWO PREDICTOR FUNCTIONS ARE RANDOMLY GENERATED FOR EACH GENE.

	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
	$n = 2$				$n = 3$			
Mem_{MCA} (Byte)	0.0026M	0.0068M	0.0156M	0.0314M	0.0183M	0.0968M	0.3630M	1.0767M
Avg. time (s)	0.016257	0.020486	0.016485	0.049608	0.043657	0.093971	0.383156	1.050769
Mem_{SMN} (Byte)	5.9797M	5.9798M	5.9799M	5.9799M	8.7265M	8.7268M	8.7272M	8.7279M
Avg. time (s)	0.465250	0.454927	0.456008	0.473981	0.694176	0.724946	0.695185	0.690967
	$n = 4$				$n = 5$			
Mem_{MCA} (Byte)	0.1544M	1.5109M	8.9656M	38.4939M	1.3628M	24.0411M	223.6387M	1.3843×10^3M
Avg. time (s)	0.243478	1.629470	8.522092	35.325539	1.402525	21.860890	200.207121	1272.674693
Mem_{SMN} (Byte)	11.4735M	11.4748M	11.4777M	11.4828M	14.2214M	14.2273M	14.2433M	14.2788M
Avg. time (s)	0.927799	0.927331	0.924562	0.934311	1.174922	1.186381	1.198102	1.290500

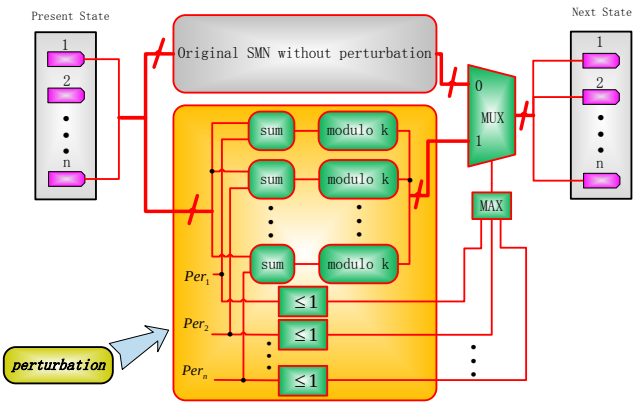

Fig. 4. An SMN with perturbation. Gene perturbation is implemented by the sum and modulo k functions of the perturbation vector and the present state.

genes, if n_0 genes are to be perturbed, this indicates that the perturbation flag vector $\boldsymbol{\gamma}$ contains n_0 non-zero values and $n - n_0$ zeros. For each zero, the current state of the corresponding gene remains, as shown in the aforementioned example. For the n_0 non-zero values, a different set of values leads to a different next state of the perturbed genes. For random gene perturbation, each set occurs with a probability of $[1/(k-1)]^{n_0}$, so the network transition from the present state to a particular next state, i.e. $\boldsymbol{S}^{(t)} \to \boldsymbol{S}^{(t+1)}$, occurs with a probability of $p^{n_0} \cdot (1-p)^{n-n_0} \cdot [1/(k-1)]^{n_0}$. Since a perturbed state is considered to be different from the present state, i.e. $\boldsymbol{S}^{(t+1)} \neq \boldsymbol{S}^{(t)}$, under perturbation, the probability of the state transition of (8) is given by (5).

To account for the perturbation effect, a modified SMN is shown in Fig. 4. The probability that the multiple-valued network is left without perturbation or that a perturbation takes effect, is determined by the output of an n-input MAX gate.

In the SMN in Fig. 4, gene perturbation is considered as follows. Since a random gene perturbation probabilistically changes the state of a gene, the modules of sum and modulo k operations are used to implement the perturbation function (of the perturbation vector and the genes' current states). The j^{th} perturbation vector, Per_j, consists of a number of i's, $i = 0,1,\cdots,(k-1)$; for instance, if an L-bit sequence Per_i is used to indicate the perturbation rate p in a ternary network and let $M = L \cdot p$, then there are $L - M$ 0's, $M/2$ 1's and $M/2$ 2's in the sequence.

This indicates that if a gene at state i is perturbed, the new state can be any j $(j \neq i)$ with an equal probability of $1/(k-1)$. Hence, if n_0 genes are perturbed, a perturbed state is chosen with a probability of $[1/(k-1)]^{n_0}$. The probability that either an original multiple-valued function works or a perturbation occurs (by (8)) is implemented by the output sequence of an n-input MAX gate. This sequence is then used as the control sequence of a bus (or multiple-bit) multiplexer. If no perturbation occurs, the perturbation vectors ($'Per_1'$, $'Per_2'$, $\cdots$, $'Per_n'$ in Fig. 4) consist of all 0's, and thus the output sequence of the MAX gate will contain all 0's. The next state is subsequently given by the original SMN without perturbation; otherwise, the next state is determined by the perturbation probability encoded in the output sequence of the MAX gate. From this analysis, it can be seen that the SMN model implements the function of (8) and thus computes the transition probability of (5). This indicates that it accurately implements a probabilistic multiple-valued network with perturbation.

D. State Transition Matrix and Steady State Analysis

In the simulation of an SMN, each input combination results in output sequences that contain information about the transition probability from this input to every output (or next state). For a deterministic input (i.e. the present state), the proportions of the numbers of the next states encoded in the output sequences return the statistics as the transition probabilities in a row in the state transition matrix (STM). Hence, all the transition probabilities for this input can be generated in a single run. For a probabilistic multiple-valued network (PMN) with k levels and n genes, the SMN needs to be run for each of the k^n input states and an $O(n)$ number of sequences need to be generated for the control signals of the multiplexers.

The accuracy in the computed state transition probabilities is determined by the length of the stochastic sequences. Since longer sequences are usually required in a larger network for achieving an evaluation accuracy, a factor, L, is used here to account for the computational overhead required by using a longer stochastic sequence. For a k-valued network of n genes, a complexity of $O(nLk^n)$ results for computing the STM at a desired accuracy. As shown in the simulation results in Table 2 for ternary networks, the required minimum sequence length increases slower with the numbers of genes than the number of possible networks, N, which generally increases exponentially with the number of genes in a network. Therefore, the complexity of using an SMN to compute the STM, i.e., $O(nLk^n)$, is smaller than $O(nNk^{2n})$ of an accurate analysis [24]. This difference becomes significant for a large network, as indicated by the shorter average run time in Table 2.

In a network with a large number of genes, a matrix-based analysis becomes cumbersome because of the size of the involved matrices. A steady state analysis becomes even more challenging. Using an SMN, however, the STM can be accurately and efficiently computed. The steady state distribution (SSD) can be evaluated by using the so-called time-frame expansion technique [9].

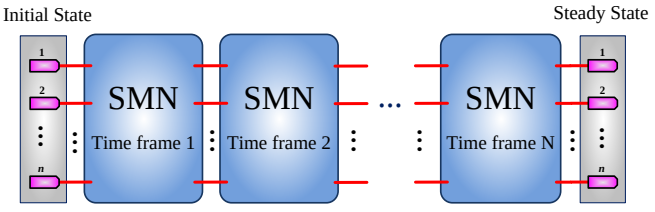

Fig. 5. A time-frame expanded SMN.

By this technique, the temporal evolution of a multiple-valued network is simulated using a spatially iterative structure of the SMN, as shown in Fig. 5. The number of iterations is determined by the number of state transitions before reaching a steady state.

A general multiple-valued network (with any *k*) can be analyzed by the time-frame expanded SMN approach. The simulation results in Table 3 reveal that, while the SMN approach takes longer time than a Markov chain analysis [24] for small networks, it becomes faster in the analysis of large networks. Although the evaluation accuracy slightly decreases with the increase of the discretization level, *k*, a better accuracy is obtained when longer stochastic sequences are used.

The memory usage of the SMN approach is further investigated and compared to that of the Markov chain analysis [24]. As shown in the simulation results in Table 4, the Markov chain analysis requires less memory than SMN for small networks with a low quantization level, *k*, whereas the required memory outgrows that of the SMN approach in the analysis of a larger network with a larger *k*. In fact, the required memory by the Markov chain analysis increases exponentially with the number of genes and depends heavily on *k*, because of the increased size of transition matrices in an analysis. On the other hand, the memory required by the time frame expanded SMN approach is mainly determined by the sequence length (*L*) and number of genes (*n*), while the quantization level (*k*) has little impact. It is also shown that the Markov chain analysis incurs a significantly longer run time than the SMN approach in the analysis of networks with larger *n* and *k*. Although a constant sequence length (30K) is used for the simulation results in Table 4, further simulations using different sequence lengths show a similar pattern. As reported in the Results and Discussion section, these features make the SMN approach more efficient than an analytical Markov chain approach while producing very accurate results compared to the Monte Carlo method in the analysis of large gene networks.

IV. Results and Discussion

A. A Multiple-valued p53-Mdm2 Network

p53 is a tumour suppressor gene that plays an important role in preventing the development and progression of tumour cells [31, 32]. External stimuli such as DNA damages can activate signaling pathways that involve the genes p53 and Mdm2. The dynamic behavior of a p53 network has been studied by using various Boolean models [18, 33] and an oscillatory behavior of the p53 and Mdm2 has been observed [10, 34].

A four-node network has been analyzed in [18, 19] with "DNA damage" as one of the nodes. As DNA damage (such as double strand breaks) is one of the major factors that activate the p53 network [10, 32, 34], a three-node network that excludes the DNA damage as an external factor, as shown in Fig. 6, is considered in this section for an application of the SMN model. Let X_1 denote the gene p53, cytoplasmic p53 and nucleic p53 (i.e. protein p53), and X_2 and X_3 denote the cytoplasmic Mdm2 and nucleic Mdm2, respectively. As protein p53 activates the cytoplasmic Mdm2 that has a positive effect on the nuclear Mdm2. Thus, protein p53 promotes nucleic Mdm2 indirectly through the cytoplasmic Mdm2. At the same time, p53 down-regulates nucleic Mdm2 by directly inhibiting the nuclear translocation of p53 [18, 19].

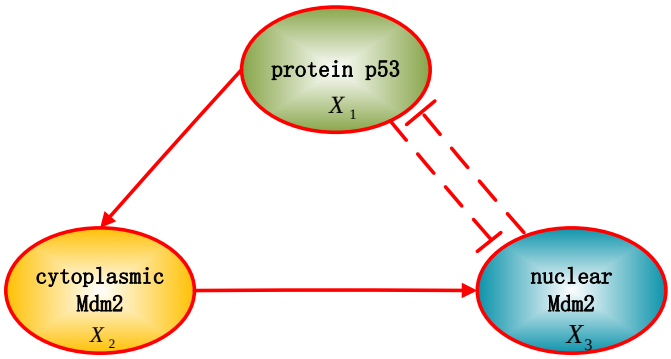

Fig. 6. The multiple-valued p53-Mdm2 network under DNA damage (adapted from [18, 19]).

Based on these interactions, an SMN for the p53 network is established as follows: $V = \{X_1, X_2, X_3\}$, where X_1 has ternary values, each of which indicates a different concentration level of the p53 protein (low, medium and high) [18], while X_2 and X_3 are binary nodes, with the ternary functional sets $F_1 = \{f_1^{(1)}, f_2^{(1)}\}$, $F_2 = \{f_1^{(2)}, f_2^{(2)}\}$, and $F_3 = \{f_1^{(3)}, f_2^{(3)}\}$. Given their truth tables [19], these functions can be implemented by multiple-valued logic gates. For the gene node X_2 (i.e. cytoplasmic Mdm2), for example, the state transitions are shown in the first and last columns in Table 5. These transitions can be implemented by an ES operator and two rotate gates, as shown in Fig. 7. The intermediate states during the transitions are shown in Table 5.

TABLE 5 STATE TRANSITIONS OF X_2

X_1	X^1 (≥1)	X^2 (rotate)	X_2
0	1	2	0
1	1	2	0
2	2	0	1

Fig. 7. A stochastic multiple-valued network for gene X_2 (cytoplasmic Mdm2).

Similarly, the implementation functions for the other genes X_1 and X_3 can be determined from their truth tables as well (in Tables 6 and 7 respectively).

TABLE 6 TRUTH TABLE FOR X_1 [19]

X_3	X_1	X_1
0	0	1
0	1	2
0	2	2
1	0	0
1	1	0
1	2	1

TABLE 7 TRUTH TABLE FOR X_3 (ADAPTED FROM [19])

X_1	X_2	X_3
0	0	0
0	1	1
1	0	0
1	1	1
2	0	0
2	1	1

While the state transition in [19] is dependent on the current state and the state after transition, random state transitions are considered in this work, as in [6-8, 24]. Under this assumption, the present state is transitioned into a next state with a transition probability when perturbation occurs. The selection probabilities are shown in Table 8 for the predictor functions.

TABLE 8 THE SELECTION PROBABILITIES OF PREDICTOR FUNCTIONS FOR THE MULTIPLE-VALUED P53-MDM2 NETWORK.

f^1	f^2	f^3
0.95	0.95	0.95
0.05	0.05	0.05

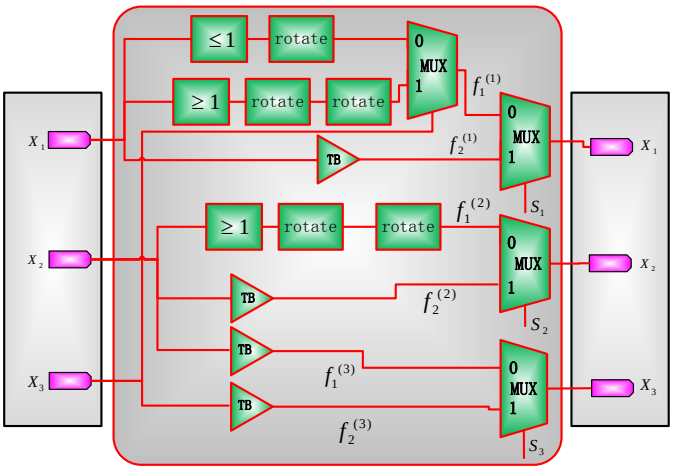

Fig. 8. A stochastic multiple-valued network (SMN) for the p53-Mdm2 network under DNA damage.

For the p53-Mdm2 network in Fig. 6, an SMN can be constructed for implementing its functions, as shown in Fig. 8. For this three-gene network, a two-input multiplexer is used for each gene to probabilistically select a function with the selection probability encoded in the control sequence. For the update functions, $f_1^{(i)}$ ($i \in \{1,2,3\}$) is for the state transition due to interactions with other genes or the change of the current state, while $f_2^{(j)}$ ($j \in \{1,2,3\}$) indicates the preservation of the current state. In this model, the effect of asynchronicity [35] is implicitly considered at each step of the state updating process. For each input state, the output sequences are read out and decoded into (transition) probabilities.

The p53 SMN model is used to compute the state transition matrix (STM) for this network, which is compared to the STM obtained by a Markov chain analysis. The norms $\|\cdot\|_1$, $\|\cdot\|_2$, and $\|\cdot\|_\infty$ are then used to measure the differences of the STMs obtained by the different methods. $\|\cdot\|_1$ and $\|\cdot\|_\infty$ indicate the maximum absolute values of the summed differences of the columns and rows respectively, while $\|\cdot\|_2$ measures the average difference of all entries. Let $\boldsymbol{A}_{SMN}$ and $\boldsymbol{A}_{MCA}$ be the STMs obtained by the SMN and a Markov chain analysis; the difference between these two matrices is then given by $\Delta\boldsymbol{A} = \boldsymbol{A}_{SMN} - \boldsymbol{A}_{MCA}$. For the multiple-valued p53-Mdm2 network with no perturbation, we obtain $\|\Delta\boldsymbol{A}\|_1$ = 0.0049, $\|\Delta\boldsymbol{A}\|_2$ = 0.0023 and $\|\Delta\boldsymbol{A}\|_\infty$ = 0.0021 by using a sequence length of 10,000 values for the SMN.

The STM of the p53-Mdm2 network under perturbation can similarly be computed using an SMN with perturbation (by implementing the SMN in Fig. 8 into that of Fig. 4). The STMs obtained by different approaches are illustrated in Fig. 9, while the norms of the differences, $\|\Delta\boldsymbol{A}\|_1$, $\|\Delta\boldsymbol{A}\|_2$ and $\|\Delta\boldsymbol{A}\|_\infty$, are shown in Table 9 for using different sequence lengths. The average run time is also provided for both approaches.

As revealed in Table 9, the difference between the STMs computed using the SMN and the Markov chain analysis decreases with the increase of sequence length L. For the same accuracy requirement, as can be seen, a larger sequence length is needed for a higher perturbation rate. This relationship between the sequence length and perturbation rate is further shown in Fig. 10. However, the computational inaccuracy due to the inherent stochastic fluctuation in stochastic computation is generally small and negligible. Hence, the proposed SMN approach can accurately and efficiently compute the STM of a probabilistic multiple-valued network (PMN) with or without perturbation.

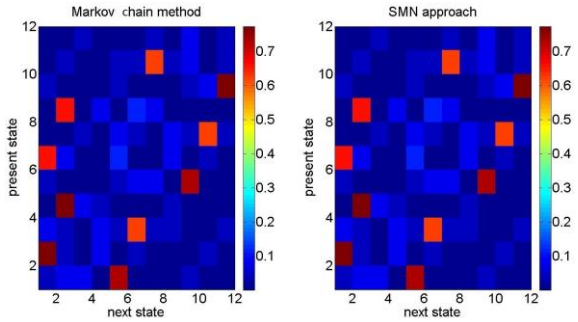

Fig. 9. State transition matrices (STMs) obtained by the Markov chain [24] and SMN approaches for the p53-Mdm2 network. Sequence length: L = 10,000 bits; perturbation rate: p = 0.1.

TABLE 9 NORMS OF THE DIFFERENCE BETWEEN THE STMS OBTAINED BY MARKOV CHAIN ANALYSIS (MCA) AND THE SMN APPROACH FOR THE P53-MDM2 NETWORK, $\Delta\boldsymbol{A}_{MCA-SMN}$. p: PERTURBATION RATE; L: SEQUENCE LENGTH FOR THE STOCHASTIC APPROACH.

		$p = 0$		
L (bits)		1,000	10,000	100,000
$\|\Delta\boldsymbol{A}_{MCA-SMN}\|_1$		0.0091	0.0049	7.6500×10^{-4}
$\|\Delta\boldsymbol{A}_{MCA-SMN}\|_2$		0.0091	0.0023	8.1496×10^{-4}
$\|\Delta\boldsymbol{A}_{MCA-SMN}\|_\infty$		0.0183	0.0021	0.0016
Average time (s)	MCA	0.00522		
	SMN	0.06804	0.57853	5.72595
		$p = 0.1$		
L (bits)		1,000	10,000	100,000
$\|\Delta\boldsymbol{A}_{MCA-SMN}\|_1$		0.0368	0.0097	0.0030
$\|\Delta\boldsymbol{A}_{MCA-SMN}\|_2$		0.0210	0.0061	0.0016
$\|\Delta\boldsymbol{A}_{MCA-SMN}\|_\infty$		0.0401	0.0105	0.0032
Average time (s)	MCA	0.01538		
	SMN	0.05937	0.64545	5.96927

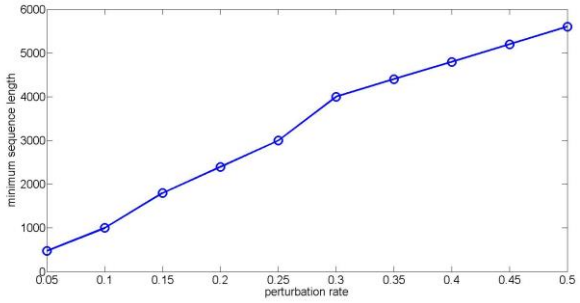

Fig. 10. The relationship between the minimum sequence length required for computing the STM (with an accuracy requirement of $\|\cdot\|_2 = 0.02$) and the perturbation rate for the multiple-valued p53-Mdm2 network.

A probabilistic network with random perturbation evolves as an ergodic Markov chain [7], because the non-zero perturbation rate makes all the states in the network connected. Hence, a steady state exists in a network with perturbation. The steady state distribution (SSD) for the p53 network under DNA damage is obtained by using different approaches, as shown in Fig. 11.

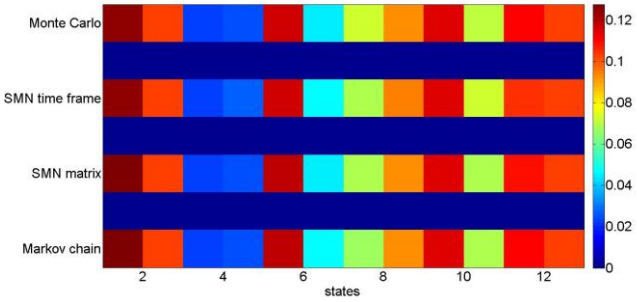

Fig. 11. Steady state distributions (SSDs) of the multiple-valued p53 network after 30 state transitions with an initial state of 000. The X-axis indicates the network state, and the Y-axis is for the different approaches. The color bar on the right shows the values of the SSD. Perturbation rate: $p = 0.1$; sequence length or simulation runs: 10,000.

As shown in Fig. 11, all approaches produce similar SSDs. In fact, the difference between the results by the SMN and the accurate Markov chain analysis is negligible when reasonably long stochastic sequences are used (such as those of 10,000 values). Using the STM computed by an SMN approach or the time frame expanded SMN approach results in a very accurate approximation of the SSD compared to the rigorous Markov chain analysis. A further analysis shows that the relative error is less than approximately 0.2% for the stochastic approach. Individual gene expressions are shown in Fig. 12 for a single simulation of 30 transitions. It can be seen that the likely expression levels of p53 and nuclear Mdm2 follow an oscillatory pattern as analytically [19] and experimentally [36] shown previously.

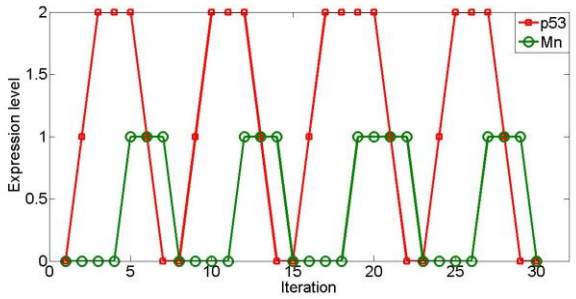

Fig. 12. Individual gene expressions for the p53 network generated from a single simulation of 30 iterations with an initial state of 011. X-axis indicates the iteration number and Y-axis shows the expression level of p53 or nuclear Mdm2.

B. A WNT5A network

Next, a WNT5A network [24] is used to illustrate the efficiency of the stochastic multiple-valued network (SMN) model and the time-frame expansion technique. A ten-gene network is derived from the predictive relationships in Table 10. The selection probabilities of the predictor functions are also given in Table 10 (estimated from [24]). Fig. 13 shows a detailed structure of the network with double (or single) - headed arrows indicating the bi (or uni) - directional relationships of gene pairs. While the number of output arcs varies, every node (or gene) has three input arcs in Fig. 14.

TABLE 10 THE SELECTION PROBABILITY OF PREDICTOR FUNCTIONS FOR A 10-GENE WNT5A NETWORK (ESTIMATED FROM [24]).

Target	Predictor f_1	Select prob.	Predictor f_2	Select prob.	Predictor f_3	Select prob.
pirin	WNT5A	0.6	STC2	0.2	HADHB	0.2
WNT5A	pirin	0.6	S100P	0.2	RET-1	0.2
S100P	WNT5A	0.33	RET-1	0.33	Synuclein	0.34
RET-1	pirin	0.43	WNT5A	0.24	S100P	0.33
MMP-3	S100P	0.43	RET-1	0.25	HADHB	0.32
PHO-C	MART-1	0.33	Synuclein	0.33	STC-2	0.34
MART-1	pirin	0.44	WNT5A	0.28	MMP-3	0.28
HADHB	pirin	0.3	WNT5A	0.4	MMP-3	0.3
Synuclein	pirin	0.25	S100P	0.25	MART-1	0.5
STC2	pirin	0.35	WNT5A	0.3	PHO-C	0.35

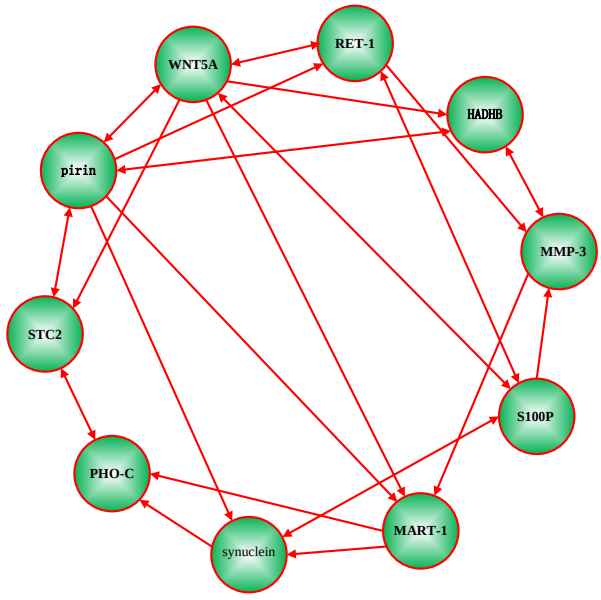

Fig. 13. A ternary WNT5A network with gene interactions (adapted from [24]).

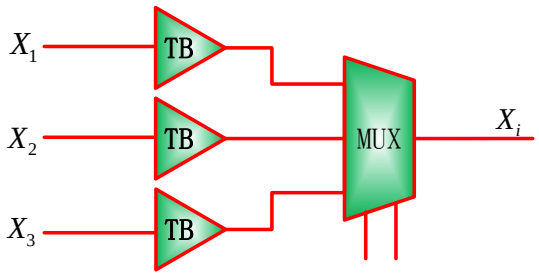

Fig. 14. An SMN module for certain gene i of the ternary WNT5A network, with the predictor function implemented by a ternary buffer. Let $\boldsymbol{G}_i = (X_1, X_2, X_3)$ be the input vector for gene i; the input vector for each of the genes in the ternary WNT5A network is given by:
$\boldsymbol{G}_{WNT5A} = (pirin, S100P, RET-1)$;
$\boldsymbol{G}_{pirin} = (WNT5A, STC2, HADHB)$;
$\boldsymbol{G}_{RET-1} = (pirin, WNT5A, S100P)$;
$\boldsymbol{G}_{HADHB} = (pirin, WNT5A, MMP-3)$;
$\boldsymbol{G}_{MMP-3} = (S100P, RET-1, HADHB)$;
$\boldsymbol{G}_{S100P} = (WNT5A, RET-1, Synuclein)$;
$\boldsymbol{G}_{MART-1} = (pirin, WNT5A, MMP-3)$;
$\boldsymbol{G}_{Synuclein} = (pirin, S100P, MART-1)$;
$\boldsymbol{G}_{PHO-C} = (MART-1, Synuclein, STC2)$;
$\boldsymbol{G}_{STC2} = (pirin, WNT5A, PHO-C)$.

For the 10-gene ternary WNT5A network, it requires a state transition matrix (STM) of $3^{10} = 59049$ columns and rows for an accurate analysis. This makes it difficult, if not impossible, to estimate the steady state of an SMN using a matrix-based analysis. In general, it is difficult to analyze a large gene network, due to its excessive computational overhead. A Monte Carlo (MC) method has been used in [24] for evaluating the steady state distribution (SSD) of a network with perturbation. However, the MC method is very time consuming due to the slow convergence typically encountered in an MC simulation.

However, an SMN model can be constructed for the ternary WNT5A network, as shown in Fig. 14. For this SMN, the SSD can be estimated using the aforementioned time-frame expansion technique and compared with the MC simulation [24]. By the time-frame expansion technique, the temporal operation of an SMN is laid out into a series of identical SMN modules in the spatial domain (as in Fig. 5). The required iterations of the SMN are determined by the number of state transitions before reaching a steady state. As in [37], a steady state is considered to have been reached if the discrepancy between two adjacent simulations is smaller than a threshold or the number of simulations has reached a maximum value. The state or GAP of the WNT5A network can be represented by a ternary vector as $(x_1, x_2, \cdots, x_{10})$, or its decimal index. The SSDs of the network with all of the 59049 states, obtained using the SMN and the MC method [24], are shown in Fig. 15.

The norms of the differences of the SSDs obtained using the time frame expanded SMN approach with different sequence lengths and the MC method are shown in Table 11. As can be seen, the time-frame expanded SMN technique efficiently evaluates the SSD of the WNT5A network and produces very accurate results compared to the Monte Carlo simulation [24]. The average run time reveals the efficiency of the SMN approach. This is because the use of randomly permuted sequences results in a faster convergence than in the MC simulation. The use of longer stochastic sequences further improves the accuracy of evaluation and remains more efficient by several orders of magnitude than the MC method. Albeit at a higher memory cost than the MC simulation (shown in Table 11), the SMN approach requires much less memory than an accurate approach such as a Markov chain analysis (shown in Table 4). Since it is difficult to compute the STM or SSD of a large GRN by using an accurate analysis, a time-frame expanded SMN provides an alternative method to evaluate the SSD of a large network with a tunable tradeoff between accuracy and efficiency by using stochastic sequences of different lengths.

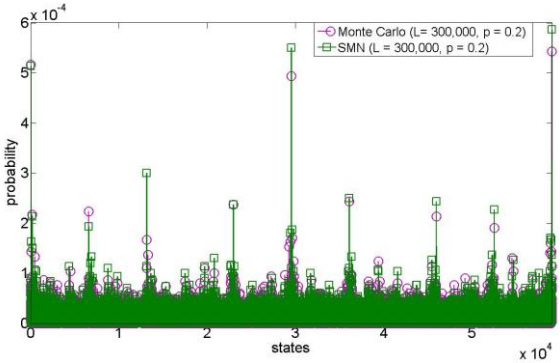

Fig. 15. SSDs of the ternary WNT5A network using the SMN model and Monte Carlo (MC) simulation with perturbation rate $p = 0.2$ and sequence length or simulation runs $L/Num = 300{,}000$.

TABLE 11 NORMS OF THE DIFFERENCE BETWEEN THE SSDS OBTAINED BY THE TIME FRAME EXPANDED SMN TECHNIQUE AND MONTE CARLO (MC) SIMULATION FOR THE TERNARY WNT5A NETWORK WITH PERTURBATION RATE $p = 0.2$. THE AVERAGE RUN TIME IS ALSO SHOWN. L: SEQUENCE LENGTH FOR THE STOCHASTIC APPROACH; Num: NUMBER OF SIMULATION RUNS FOR THE MC METHOD; $\boldsymbol{SSD}_{MC}$ AND $\boldsymbol{SSD}_{SMN}$ RESPECTIVELY DENOTE THE STEADY STATE DISTRIBUTIONS OBTAINED BY THE MC SIMULATION AND THE TIME FRAME EXPANDED SMN TECHNIQUE; A MAXIMUM NUMBER OF 50 ITERATIONS IS APPLIED TO THE STEADY STATE EVALUATION.

Num/L		3k	30k	300k	3000k
$\|\boldsymbol{SSD}_{MC} - \boldsymbol{SSD}_{SMN}\|_1$		1.8827	1.3291	0.4915	0.1605
$\|\boldsymbol{SSD}_{MC} - \boldsymbol{SSD}_{SMN}\|_2$		0.0258	0.0082	0.0026	8.5342×10^{-4}
$\|\boldsymbol{SSD}_{MC} - \boldsymbol{SSD}_{SMN}\|_\infty$		1.0000×10^{-3}	2.6667×10^{-4}	1.3333×10^{-4}	5.6333×10^{-5}
Average time (s)	MC	98.4768	981.159	9731.04	97336.5
	SMN	0.47811	4.23694	58.9336	673.928
Required memory (M Byte)	MC	2.7117	10.0076	51.2108	599.607
	SMN	9.8083	40.9368	373.299	3696.5

V. CONCLUSION

As a generalization of stochastic Boolean networks (SBNs), stochastic multiple-valued networks (SMNs) are proposed as an efficient approach to modeling the effects of noise in gene regulatory networks (GRNs). In an SMN, the state transition matrix can be accurately and efficiently computed with a complexity of $O(nLk^n)$, where n is the number of genes in a network, k is the quantization level of a gene's state and L is a factor determined by the stochastic sequence length. Since L increases slower with n than the number of network functions N, this result is an improvement compared to the previous

result of $O(nNk^{2n})$ for an accurate analysis. The use of randomly permuted sequences further increases computational efficiency and allows for a tunable tradeoff between accuracy and efficiency. A steady state analysis using a time-frame expansion technique has shown a significant speedup compared to an accurate Markov chain analysis and produced very accurate results compared to Monte Carlo simulation.

SMNs are constructed for the analysis of a multiple-valued p53-Mdm2 network and a ternary WNT5A network under gene perturbation. Simulations of the SMNs have revealed the oscillatory dynamics of the p53-Mdm2 network with random gene perturbation. The SMN approach can also efficiently predict the steady state distribution of the WNT5A network with gene perturbation. Hence, the SMNs are useful in evaluating the effects of gene perturbation and, potentially, helpful in drug discovery for an intervention-based gene therapy. Future work includes the further investigation of asynchronous networks [38], as well as stochastic networks with time delays [39] and parameter uncertainties [40].

REFERENCES

[1] Elowitz, M. B., Levine, A. J., Siggia, E. D. & Swain, P. S.: Stochastic gene expression in a single cell. Science 297, 1183–1186 (2002).

[2] Karlebach G, Shamir R.: Modelling and analysis of gene regulatory networks. Nat Rev Mol Cell Biol 2008, 9:770-780.

[3] Kauffman S. A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Theor. Biol., vol. 22, pp. 437–467, 1969.

[4] Glass, L. and Kauffman, S.: The logical analysis of continuous non-linear biochemical control networks. J. Theor. Biol., 39, 103–129, 1973.

[5] Huang, S.: Gene expression profiling, genetic networks, and cellular states: An integrating concept for tumorigenesis and drug discovery. J. Mol. Med., 77,469–480, 1999.

[6] Shmulevich, I., Dougherty E. R., and Zhang. W.: From Boolean to probabilistic Boolean networks as models of genetic regulatory networks. Proc. IEEE, vol. 90, pp. 1778–1792, 2002(a).

[7] Shmulevich, I., Dougherty, E.R. and Zhang, W.: Gene perturbation and intervention in probabilistic Boolean networks. Bioinformatics, 2002(b), 18(10):1319-1331.

[8] Shmulevich, I., Dougherty, E.: Probabilistic Boolean Networks: The Modeling and Control of Gene Regulatory Networks. SIAM, Philadelphia (2009).

[9] Liang, J. and Han, J.: Stochastic Boolean Networks: An Efficient Approach to Modeling Gene Regulatory Networks. BMC Systems Biology, 6:113, 2012.

[10] Lahav G., Rosenfeld N., Sigal A., Geva-Zatorsky N., Levine AJ., Elowitz MB., Alon U.: Dynamics of the p53-Mdm2 feedback loop in individual cells. Nat Genet 2004, 36:147–150.

[11] Martin, S., Zhang, Z., Martino, A. and Faulon, J-L.: Boolean dynamics of genetic regulatory networks infer red from microarray time series data. Bioinformatics, 2007, 23(7): 866-874.

[12] Harvey, I. & Bossomaier, T.: Time out of joint: Attractors in asynchronous random Boolean networks. Proc. Fourth European Conf. Artificial Life (ECAL97) , eds. Husbands, P. & Harvey, I. (MIT Press), pp. 67–75, 1997.

[13] Kitano, H.: Foundations of Systems Biology (MIT Press Cambridge, Massachusetts London), 2001.

[14] Thomas R. and D'Ari R., Biological Feedback. CRC Press, 1990.

[15] Morris M. K., Saez-Rodriguez J., Sorger P. K., Lauffenburger D. A.: Logic-based models for the analysis of cell signaling networks. Biochemistry 49, 3216–3224 (2010).

[16] Dubrova, E.: Random Multiple-Valued Networks: Theory and Applications. Proc. Int'l Symp. Multiple-Valued Logic (ISMVL '06), pp. 27-33, May 2006.

[17] Garg A., Mendoza L., Xenarios I., and DeMicheli G.: Modeling of Multiple Valued Gene Regulatory Networks. Proc. 29th IEEE Int'l Conf. Eng. In Medicine and Biology Soc. (EMBC '07), pp. 1398-1404, Aug. 2007.

[18] Abou-Jaoude W., Ouattara D., Kaufman M.: From structure to dynamics: frequency tuning in the p53-mdm2 network: I. logical approach. J Theor Biol. 258(4), 561–577 (2009). doi:10.1016/j.jtbi.2009.02.005.

[19] Murrugarra, D., Veliz-Cuba, A., Aguilar, B., Arat, S., & Laubenbacher, R.: Modeling stochasticity and variability in gene regulatory networks. EURASIP J. Bioinform. Syst. Biol., 2012 (1), 5.

[20] Li Z., Cheng D.: Algebraic Approach To Dynamics of Multivalued Networks. International Journal of Bifurcation and Chaos, Vol. 20, No. 3 (2010) 561– 582.

[21] Adamatzky A.: On dynamically non-trivial three-valued logics: Oscillatory and bifurcatory species. Chaos Solit. Fract.18, 917–936, 2003.

[22] Volker, L. G. & Conrad, M.: The role of weak interactions in biological systems: The dual dynamic model. J. Theor. Biol. 193, 287–306，1998.

[23] Aldana M., Coopersmith S., and Kadanoff L. P.: Boolean dynamics with random couplings. http://arXiv.org/abs/adap-org/9305001.

[24] Kim S., Li H., Dougherty E. R., et al.: Can Markov chain models mimic biological regulation? Journal of Biological Systems, vol. 10, no. 4, pp. 337–357, 2002.

[25] Shmulevich, I. et al. .: Steady-state analysis of genetic regulatory networks modeled by probabilistic Boolean networks. Comp. Funct. Genomics, 4, 601–608, 2003.

[26] Rosenthal, J. S.: Minorization conditions and convergence rates for Markov chain Monte Carlo. J. Am. Stat. Assoc., 90, 558–566, 1995.

[27] Ching W., Zhang S., NG M., Akutsu T.: An approximation method for solving the steady-state probability distribution of probabilistic Boolean networks. Bioinformatics, 2007, 23, pp. 1511 –151.

[28] Guelzim N., Bottani S., Bourgine P., Kepes F.: Topological and causal structure of the yeast transcriptional regulatory network. Nat Genet 2002, 31:60–63.

[29] Gaines, B.R.: Stochastic Computing Systems. Advances in Information Systems Science, Vol. 2, pp. 37-172, 1969.

[30] Han J., Chen H., Liang J., Zhu P., Yang Z. and Lombardi F.: A Stochastic Computational Approach for Accurate and Efficient Reliability Evaluation, IEEE Transactions on Computers, in press, 2013.

[31] Weinberg RA.: The Biology of Cancer. 1st edition. New York: Garland Science; 2006.

[32] Vogelstein B, Lane D, Levine AJ.: Surfing the p53 network. Nature 2000, 408:307–310.

[33] Ciliberto, A., Novak B. and Tyson J.J.: Steady states and oscillations in the p53-Mdm2 network. Cell Cycle, 4: 486-493, 2005.

[34] Batchelor, E., Loewer, A. and Lahav, G.: The ups and downs of p53: understanding protein dynamics in single cells. Nature Reviews Cancer 2009, 371-377.

[35] Luo C., Wang X. (2013): Dynamics of Random Boolean Networks under Fully Asynchronous Stochastic Update Based on Linear Representation. PLoS ONE 8(6): e66491. doi:10.1371/journal.pone. 0066491

[36] Geva-Zatorsky, N., Rosenfeld, N., Itzkovitz, S., Milo, R., Sigal A., Dekel E., Yarnitzky T., Liron Y., Polak P., Lahav G., Alon U.: Oscillations and variability in the p53 system. Mol Syst Biol. 2. 2006.0033, (2006) doi: 10.1038/msb4100068.

[37] Zhang, S. et al.: Simulation study in probabilistic Boolean network models for genetic regulatory networks. Int. J. Data Min. Bioinformatics, 1, 217–240, 2007.

[38] Garg A., Di Cara A., Xenarios I., Mendoza L., De Micheli G.: (2008) Synchronous versus asynchronous modeling of gene regulatory networks. Bioinformatics 24, 1917– 1925.

[39] Chen, B.S. and Chen, P.W. (2008): Robust engineered circuit design principles for stochastic biochemical networks with parameter uncertainties and disturbances. IEEE Trans. Biomed. Circuits Syst., 2, 114–132.

[40] Wu, F. X.: Global and robust stability analysis of genetic regulatory networks with time-varying delays and parameter uncertainties, IEEE Trans. Biomed. Circuits Syst., vol. 5, no. 5, pp. 391–398, 2011.