Evaluation of attractors and basins of asynchronous random Boolean networks

Meng Yang and Tianguang Chu[*]

State Key Laboratory for Turbulence and Complex Systems, College of Engineering, Peking University, Beijing 100871, China

(Received 21 June 2011; revised manuscript received 8 September 2011; published 4 May 2012)

We present an algebraic approach for determining the attractors and their basins of random Boolean networks under an asynchronous stochastic update based on the recently developed matrix semitensor product theory, which allows for converting the logical dynamics of a Boolean network into a standard iterative dynamics. In this setting, all attractors and basins are specified by the network transition matrices. We then devise procedures that can find all attractors and their basins exactly. We also discuss the issue of overlapping basins in asynchronous random Boolean networks, and we propose methods to compute the weight of each attractor and the basin entropy of the systems.

DOI: 10.1103/PhysRevE.85.056105 PACS number(s): 89.75.Hc, 89.75.Fb, 05.45.−a

I. INTRODUCTION

A Boolean network (BN) is often used as a generic model for the dynamics of complex systems of interacting elements. The study of random Boolean networks (RBNs) was initiated by Kauffman as a model for gene regulation [1]. It is also a candidate for representing a class of behaviors observed in large regulatory networks [2]. A RBN is typically formulated as a directed graph with genes or elements, each of which is identified by a node whose state indicates whether the gene is switched on or off. Each node receives inputs from its neighbors or itself and updates its state according to a Boolean function of all inputs simultaneously. Yet there are also many real-world systems which do not evolve according to a globally synchronized clock, and imposing one may lead to spurious order [3]. It has been found that deviations from a synchronous update modify the attractors of the dynamics considerably [4,5]. To clarify the effects of the updating scheme, asynchronous random Boolean networks (ARBNs) have received an increasing amount of interest in recent years [6,7].

For BNs, the states that are revisited infinitely often in the long-time limit starting from a random initial condition constitute an attractor, which can be in the form of either a single state (fixed point) or a repeating set of states (cycle), and the number of states on an attractor is called the size of the attractor. Since the total number of states that a network of N nodes can have is finite (2^N), after a large enough number of time steps the system must necessarily return to a previously visited state. Transient states that lead into an attractor and the attractor itself form the basin of attraction (or basin for short) of the attractor. The cardinality of the basin of an attractor normalized by the size of the entire state space (2^N) defines the weight of the attractor, which indicates the probability that a randomly chosen state will flow into this very attractor. There have been numerous investigations focusing on the attractors and their basins of RBNs. See, e.g., Refs. [4,7–10] .

Essentially, a RBN is a logical system and it is hence not an easy task to investigate a RBN analytically in general. In most studies, RBNs (including ARBNs) were approached in terms of the relevant elements or nonfrozen nodes to determine the number and size of the attractors [8–12]. In general, these methods only estimate the average number and size of the attractors rather than giving the exact number of attractors and their specific basins. Recently, a new algebraic approach was proposed in Ref. [13] that is based on the semitensor product of matrices and can convert the logical dynamics of BNs into algebraic form of the standard discrete-time system described by difference equations. In this setting, all of the attractors and their basins of deterministic BNs can be calculated directly.

For the case of ARBNs with arbitrary connectivity, however, how to determine all attractors and their basins still remains an open question because of the very different nature of ARBNs from deterministic BNs and RBNs. Actually, the number of attractors will change when going from the synchronized RBNs to ARBNs [6]. In RBNs, all states within a given basin reach the same attractor in the end, whereas the transient states of different basins in ARBNs may overlap. These render the trajectories of the states of attractors of ARBNs much more complex and irregular than the case of RBNs, and they make it hard to find the basins of all attractors in ARBNs. Existing studies based on extensive numerical methods can locate the attractors and determine the chance of reaching each attractor in ARBNs, but they cannot obtain the basins [7].

In this paper, we are concerned with the ARBNs where a single randomly selected element is updated at each time step. By means of the matrix expression of logical functions based on the semitensor product theory, which has only been used in deterministic BNs, we present a systematic approach to find the attractors and their basins under an asynchronous stochastic update. Compared with the existing methods, our approach is completely based on analytical analysis, suggesting rigorous numerical procedures for finding all attractors and all the exact basins. We will also discuss how to deal with the overlapping parts between different basins.

II. MODEL

Consider an ARBN composed of N nodes that receive inputs from K distinct nodes. The state of node i at time t is denoted by $A_i(t)$ and the state of the jth input of node i

[*]chutg@pku.edu.cn

is denoted by $A_{i_j}(t),\ j \in \{1,\ldots,K\}$. If node i is updated at a certain time step $t+1$, the update schedule of the whole network can be described as

$$\begin{aligned} A_i(t+1) &= f_i[A_{i_1}(t),\ldots,A_{i_K}(t)], \\ A_j(t+1) &= A_j(t),\ j \in \{1,\ldots,N\}, \quad j \neq i. \end{aligned} \tag{1}$$

Following [13], we can convert the logical dynamics described in Eqs. (1) into an equivalent algebraic form of conventional linear iterative dynamics. To do this, we need the following basic notions and results. Given an $m \times n$ matrix A, a $p \times q$ matrix B, and the least common multiplier of n and p denoted by l, the semitensor product (STP) of A and B is defined as

$$A \ltimes B := (A \otimes I_{l/n})(B \otimes I_{l/p}),$$

where $\otimes$ refers to the Kronecker product of two matrices and I_m to an $m \times m$ identity matrix. The STP of matrices is a generalization of a conventional matrix product. All the fundamental properties of the conventional matrix product remain true. More details on the semitensor product and its application can be found in Ref. [13].

The state of each node in a BN can be regarded as a logical variable taking its value from a logical domain $D = \{1,0\}$, where 1 and 0 represent "true" and "false", respectively. We identify the two logical values with two vectors as $T \sim 1 \sim \delta_2^1$, $F \sim 0 \sim \delta_2^2$, where δ_n^r denotes the rth column of the identity matrix I_n. Let $\Delta_n := \{\delta_n^r | 1 \leqslant r \leqslant n\}$; for notational ease, denote this as $\Delta := \Delta_2$. Then $\Delta \sim D$. An $n \times s$ matrix M is called a logical matrix if $M = [\delta_n^{i_1}, \delta_n^{i_2}, \ldots, \delta_n^{i_s}]$, which is also briefly denoted as $M = \delta_n[i_1, i_2, \ldots, i_s]$. The set of $n \times s$ logical matrices is denoted by $\mathcal{L}_{n\times s}$.

Let f_i be the K-ary logical function in Eqs. (1). According to Ref. [13], there exists a unique $M_i \in \mathcal{L}_{2\times 2^N}$, called the structure matrix, such that

$$f_i\left[A_{i_1}(t), A_{i_2}(t), \ldots, A_{i_K}(t)\right] = M_i \ltimes_{i=1}^{N} A_i(t) = M_i x(t),$$
$$A_i(t) \in \Delta,$$

where $x(t) = \ltimes_{i=1}^{N} A_i(t) = A_1(t) \ltimes \cdots \ltimes A_N(t) \in \Delta_{2^N}$ denotes the state of all nodes at time t, which contains the information of the value of the whole network at a given time t. Then, Eqs. (1) are converted into algebraic form:

$$\begin{aligned} A_i(t+1) &= M_i x(t), \\ A_j(t+1) &= A_j(t),\ j \in \{1,\ldots,N\}, \quad j \neq i. \end{aligned} \tag{2}$$

Multiplying all the N equations of Eqs. (2) yields

$$x(t+1) = L_i x(t),\ \forall\, i \in \{1,\ldots,N\}, \tag{3}$$

where $L_i \in \mathcal{L}_{2^N \times 2^N}$ is uniquely determined by M_i for each i and is called the network transition matrix of Eqs. (1). This gives an equivalent expression of the ARBN (1) in the form of conventional discrete-time linear systems. Note that for a given ARBN, L_i can be calculated directly from the logical functions, and each L_i may be chosen randomly from the N possible options at each time step. Next, we will focus on Eq. (3).

III. FIXED POINTS

We begin by considering deterministic BNs. In this case, Eq. (3) reduces to $x(t+1) = Lx(t)$ with L denoting the unique network transition matrix. A fixed point $x_e \in \Delta_{2^N}$ is then defined by $Lx_e = x_e$. According to Ref. [13], $\delta_{2^N}^i$ is a fixed point iff the ith diagonal element l_{ii} of L equals 1, and the number of fixed points, denoted by N_e, equals the number of i such that $l_{ii} = 1$, namely, $N_e = \mathrm{Tr}(L)$.

For the case of ARBNs as described in Eq. (3), a fixed point $x_e \in \Delta_{2^N}$ is defined by $L_i x_e = x_e$ for all i. From the above results, we know that $\delta_{2^N}^j$ is a fixed point iff there exists an index j such that for any updating node i the jth diagonal element l_{jj}^i of the transition matrix L_i equals 1, and the number of fixed points N_e equals the number of all such j when i goes from 1 to N. Thus, one can find all fixed points of the ARBN (3) by computing the network transition matrices L_i.

IV. CYCLES

Unlike the case of fixed points, it is more difficult to determine all cycles of ARBNs because a state in a cycle is driven into another state randomly chosen from a subset of the cycle at the next time step. By the properties of STP [13], it is not hard to verify the following sufficient condition for cycles in an ARBN (3): If there exists a state set of s ($s \geqslant 2$) elements $C = \{\delta_{2^N}^{k_1}, \ldots, \delta_{2^N}^{k_s}\}$ such that $L_i \delta_{2^N}^{k_j} \in C$, $\forall\, i \in \{1,\ldots,N\}$, $\forall\, j \in \{1,\ldots,s\}$, where i denotes the node that is updated and $\delta_{2^N}^{k_j}$ is the jth element of the set, then C is a cycle of size (or length) s.

Based on this condition, we can devise a procedure to find all cycles for ARBNs. The basic idea is to search the cycles successively from small ones to larger ones in a reducing feasible set of finite elements. Initially, take the whole state space of the network $\{\delta_{2^N}^1, \ldots, \delta_{2^N}^{2^N}\}$ as the feasible set F, and set the size of cycles to be searched as $s = 2$. Search s states from the feasible set F that satisfy the above sufficient condition, then calculate the basin of this cycle (by the method to be given later) and remove it from F. Repeat the process until all the cycles of size s have been found. Check whether the feasible set F is empty. If "not", set $s = s+1$ and go back to find all the basins of cycles of size s. Otherwise, all the attractors and basins have been identified and the process stops.

V. BASINS

The determination of basins of attractors is crucial to an understanding of the global dynamics of BNs. As the basin of an attractor consists of all states moving toward it, we can find the basin in a backstepping manner from a given attractor of the ARBN (3). Specifically, let $\mathrm{Col}_j(L_i)$ stand for the jth column of matrix L_i. It follows from [13] that $x(t) = \delta_{2^N}^j$ and $x(t+1) = \delta_{2^N}^k$ iff $\mathrm{Col}_j(L_i) = \delta_{2^N}^k$. Therefore, we define $P^k = \{\delta_{2^N}^j | \mathrm{Col}_j(L_i) = x(t) = \delta_{2^N}^k, \forall\, i \in N\}$ to be the further state set at the last time step for a state $\delta_{2^N}^k$. Namely, given $x(t+1) = \delta_{2^N}^k$, the possible state of the ARBN (3) at time t is an element of the set P^k.

Then, given an attractor C of size s, we can obtain its basin by the following procedure. First, initialize the basin

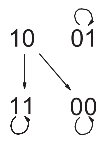

FIG. 1. The state transfer graph of an ARBN with the Boolean functions described as Eqs. (4). The transient state 10 may drive into a fixed point which is either 11 or 00. Therefore, it is the overlapping part of the basin of the fixed point 11 and that of the fixed point 00.

to be calculated, denoted by B, as $B = C$. Next, update B by replacing its elements with all the distinct elements of the union of B and P^j for all $\delta^j_{2^N} \in B, j \in \{1,\ldots,N\}$. Then, check whether the updated B thus obtained is identical with the former one. If "not", continue updating B; otherwise, the basin of attractor C is B.

At this point, we would like to remark on a unique feature of ARBNs. That is, the transient states of different basins in ARBNs may overlap. This is quite distinct from the case of RBNs. As an example, let us consider a two-node ARBN with binary state values. Suppose the Boolean functions are described as

$$A_1(t+1) = A_1(t) \wedge A_2(t), \quad A_2(t+1) = A_1(t) \vee A_2(t), \tag{4}$$

where $A_1(t)$ and $A_2(t)$ are the states of the two nodes at time t, and the state space is $\{00,01,10,11\}$. Then we get the state transfer graph as shown in Fig. 1.

Observe that the states 00,11,01 are fixed points. The transient state 10 belongs to the basins of both 11 and 00, that is, it is the overlapping part of two distinct basins. In general, the basin of an attractor may consist of overlapping and nonoverlapping parts for ARBNs. The former may drain into several distinct attractors and the latter drains into only one attractor. But in synchronized RBNs, basins never overlap. It is therefore more complex and difficult to determine basins in ARBNs.

VI. BASIN ENTROPY

As a measure of the complexity of information that a BN is capable of storing, the basin entropy is introduced in Refs. [14,15]. The basin entropy of a RBN is defined as

$$h_i = -\sum_\rho \omega_\rho \ln \omega_\rho, \tag{5}$$

where ω_ρ is the weight of attractor ρ. The average entropy over the ensemble of R realizations of RBNs is

$$\langle h \rangle = \frac{1}{R}\sum_i^R h_i. \tag{6}$$

It has been known in the literature that the basin entropy scales with the system size only in critical regimes for RBNs, thus giving a measure of complexity and criticality of RBNs. As to the case of ARBNs, little has been addressed concerning the basin entropy except for only a few numerical results [7].

A major difficulty in evaluating the basin entropy for ARBNs consists in the fact that different basins in an ARBN may overlap, as mentioned above, which complicates the task. If we define the weight ω_ρ of an attractor ρ as the size of its basin states normalized by the size of the state space (2^N) as proposed in Ref. [15], it is clear that $\sum_\rho \omega_\rho \neq 1$ in general because of the overlapping parts. A rigorous computation of ω_ρ requires accurate statistics of the weights of an overlapping part with respect to each related attractor. This can be done in the following way.

First, for the basin of an attractor ρ, identify the nonoverlapping part and all of the overlapping parts shared by other distinct basins. Construct the state transfer graph for all the n states of one of the overlapping parts ξ by representing the states as nodes in the graph. A directed link in the graph, which points from a state to its possible image state reached with weight $1/N$, indicates that a random element node is updated. An image state can be the state itself or some other state with a Hamming distance one away. If some of these images are identical, the weights of the corresponding links are added. Find all the outlinks in the graph. Here, an outlink indicates a directed link which points from a state in ξ to a nonoverlapped state in the basin of some attractor (including ρ) that shares ξ.

Next, assign the same occupation probability $1/n$ to each node in ξ initially. (The occupation probability defined here is similar to Ref. [7], but the graph concerned here is only a subgraph of the entire state space graph, and the terminational condition to be introduced below is different from that in Ref. [7].) Then, update the occupation probability of each state by dividing and moving the entire occupation probability on the node along its outgoing links according to the weights. Since an outlink only leads the occupation probability from a state in ξ to the nonoverlapping part and the states in ξ are all transient states, the occupation probability vanishes for every state in ξ but remains finite for every other state to which an outlink points. The algorithm repeats this step until the probability of each state in ξ is small enough (approaching zero), thus giving the occupation probability of each state that is pointed to by an outlink from ξ.

Finally, sum up the occupation probabilities of all states that are in the basin of ρ and pointed to by outlinks from ξ. The result is the chance to enter the basin in the long-time limit after starting from a random initial state in ξ, which is denoted by $\alpha_{\rho\xi}$. Similarly, the results for other overlapping parts of the basin of ρ can be obtained one by one.

Now, we are ready to calculate the weight of attractor ρ. Let β_ξ be the size of the overlapping part ξ normalized by the size of the state space 2^N, i.e., $\beta_\xi = n/2^N$, and let λ_ρ be the size of the nonoverlapping part of the basin of attractor ρ normalized by the size of the state space 2^N. Then we have

$$\omega_\rho = \sum_\xi \alpha_{\rho\xi}\beta_\xi + \lambda_\rho.$$

This gives a rigorous result of the weight for an attractor in ARBNs, which is identical to that obtained by the method given in Ref. [7]. Notice that this result requires finding all the outlinks of each overlapping part to different basins in a large complex network, which is usually quite computationally costly. So, it is also of interest to look for approximate but simple approaches for evaluating the attractor weights.

A simple idea is to make an even partition of an overlapping part ξ in calculation. To be specific, we can divide the size of

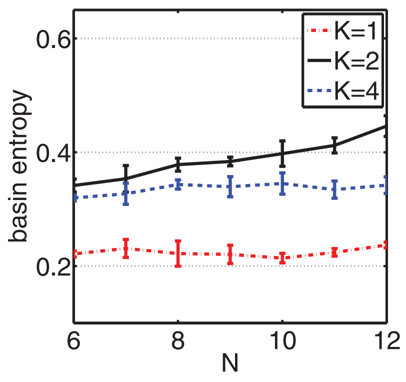

FIG. 2. (Color online) The average basin entropy for ARBNs with $K = 1,2,4$. The solid line shows an increase of $\langle h\rangle$ with N for the $K = 2$ case. The red dash-dotted line and the blue dashed line display an almost constant feature of $\langle h\rangle$ in both cases of $K = 1$ and 6. Hence, the basin entropy grows only when $K = 2$.

ξ evenly and assign the same number of states in ξ to each attractor that ξ may drain into. Then for attractor ρ, we add up the size of the nonoverlapping part and the assigned sizes of the overlapping parts of its basin, then we normalize the result based on the size of the state space to obtain an approximate value of ω_ρ. Evidently, this approach may result in certain errors in ω_ρ. We will discuss the issue later with numerical examples. Once we obtain all the weights ω_ρ, we can calculate the basin entropy $\langle h\rangle$ using Eqs. (5) and (6).

To test the effectiveness of the simple idea of an even partition in the evaluation of ω_ρ, we have calculated the average basin entropies $\langle h\rangle$ with various network sizes in ordered, critical, and chaotic phases, respectively. Figure 2 depicts the results for ARBNs with $K = 1,2,6$ and $N = 6$ to 12. It shows that the basin entropy grows with system size only in critical case ($K = 2$) for ARBNs. This agrees well with the basic finding of [7], which states that despite so many differences between ARBNs and synchronized RBNs, the basin entropy may still be regarded as a measure of complexity for ARBNs. From the results of Fig. 2 we may conclude that despite being somewhat simple and approximative, the even partition approach can lead to qualitatively reasonable results.

So far, we have discussed how to evaluate all attractors, basins, and basin entropies for a given ARBN based on its algebraic form (3). Because of the finite size of the state space, the computation will end within finite steps in principle.

VII. EXAMPLE

Let us now illustrate our approach with a simple example. Consider an ARBN network with $N = 4$ and connectivity $K = 2$ under the stochastic update rule:

$$\begin{aligned} A_1(t+1) &= A_1(t) \leftrightarrow A_2(t),\\ A_2(t+1) &= A_3(t) \vee A_2(t),\\ A_3(t+1) &= \neg[A_1(t) \wedge A_3(t)],\\ A_4(t+1) &= A_1(t) \rightarrow A_3(t), \end{aligned} \tag{7}$$

where A_i is the state of node i, $\leftrightarrow$ denotes the biconditional, $\vee$ denotes the disjunction, $\wedge$ denotes the conjunction of two propositions, $\neg$ denotes negation of one proposition, and $\rightarrow$ denotes the conditional of the first proposition to the second one. At each time step, only one equation in Eqs. (7) is picked out for an update.

The states of nodes A, B, C, and D are expressed as four-digit numbers in sequence, and the cardinal number of the state space is 2^4. It is easy to check that the attractors are a fixed point $\rho_1 = \{A_1 = 0,\ A_2 = 1,\ A_3 = 1,\ A_4 = 1\}$ (briefly denoted by $\{0111\}$) and a cycle $\rho_2 = \{1111, 1110, 1101, 1100\}$. Figures 3 and 4 are the state transfer graphs of ρ_1 and ρ_2, respectively, where the arrows denote the directed links which form the trajectories of the states. Observe that the eight states $\{1001, 0001, 1011, 0011, 1010, 1000, 0010, 0000\}$ consist of the only overlapping part ξ of the basins of the fixed point and the cycle, so $n = 8$.

We now apply our method to the system. First, we convert Eqs. (7) into the algebraic form of Eq. (3) with

$$\begin{aligned} L_1 &= [M_e(I_2 \otimes M_r)] \otimes I_4,\\ L_2 &= I_2 \otimes [M_d W_{[2]}(I_2 \otimes M_r)] \otimes I_2,\\ L_3 &= \{[I_4 \otimes (M_n M_c)]I_2 \otimes W_{[2]}M_r\} \otimes I_2,\\ L_4 &= E_d(I_{16} \otimes M_i)(I_4 \otimes W_{[2,4]})(I_2 \otimes M_r)(I_8 \otimes M_r)W_{[8,2]}, \end{aligned}$$

where M_n, M_c, M_d, M_i, M_e, and E_d denote the structure matrices of negation, conjunction, disjunction, conditional, biconditional, and dummy operator, respectively. Also, M_r is the power reducing matrix $\delta_4[1\ 4]$ such that $p^2 = M_r p$ for an arbitrary logic vector $p \in \Delta$, and $W_{[m,n]}$ is the $mn \times mn$ swap matrix such that $W_{[m,n]} \ltimes p \ltimes q = q \ltimes p$ and $W_{[n,m]} \ltimes q \ltimes p = p \ltimes q$ for any two column vectors $p \in R^m$ and $q \in R^n$

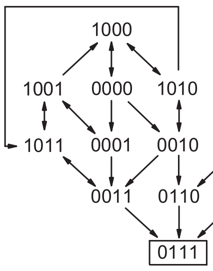

FIG. 3. The state transfer graph of the fixed point 0111, the basin of which consists of one fixed point state and eleven transient states. Among these states, the fixed point itself and the transient states $\{0110, 0101, 0100\}$ only belong to the basin of the fixed point 0111. The other eight states also belong to the basin of the cycle $\{1111, 1110, 1101, 1100\}$.

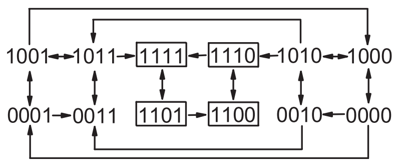

FIG. 4. The state transfer graph of the cycle $\{1111, 1110, 1101, 1100\}$, the basin of which consists of four states of the cycle and eight transient states. Note that the eight transient states also belong to the basin of the fixed point 0111. Thus, they are the overlapping part of the two basins.

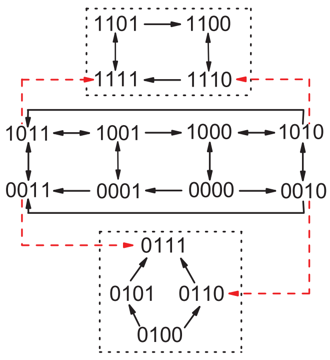

FIG. 5. (Color online) The state transfer graph of the entire state space. The states in the box form the nonoverlapping parts of basins of ρ_1 and ρ_2, respectively. The dashed arrows denote the out links which point from states in the overlapping part ξ to states in the nonoverlapping parts. The directed link for a state pointing to itself is not shown here. The weights of all the directed links in the graph take the same value: 0.25.

(see Ref. [13]). It follows immediately that

$$\begin{aligned} L_1 &= \delta_{16}[1\ 2\ 3\ 4\ 13\ 14\ 15\ 16\ 9\ 10\ 11\ 12\ 5\ 6\ 7\ 8], \\ L_2 &= \delta_{16}[1\ 2\ 3\ 4\ 1\ 2\ 7\ 8\ 9\ 10\ 11\ 12\ 9\ 10\ 15\ 16], \\ L_3 &= \delta_{16}[3\ 4\ 1\ 2\ 7\ 8\ 5\ 6\ 9\ 10\ 9\ 10\ 13\ 14\ 13\ 14], \\ L_4 &= \delta_{16}[1\ 1\ 4\ 4\ 5\ 5\ 8\ 8\ 9\ 9\ 11\ 11\ 13\ 13\ 15\ 15]. \end{aligned}$$

To find attractors and their basins, we initialize the feasible set $F = \Delta_{16}$ (the state space). It is easy to check that $\mathrm{Col}_9(L_i) = \delta_{16}^9$, $\forall\ i$. Hence, δ_{16}^9 is a fixed point. The basin of it can be calculated to be $\{\delta_{16}^k | 5 \leqslant k \leqslant 16\}$. Then, F reduces to $\{\delta_{16}^k | 1 \leqslant k \leqslant 4\}$. Similarly, we can verify that $L_i\delta_{16}^k \in \{\delta_{16}^1,\delta_{16}^2,\delta_{16}^3,\delta_{16}^4\}$, $\forall\ i,k \in \{1,2,3,4\}$. By the sufficient condition of cycles, we know that $\{\delta_{16}^1,\delta_{16}^2,\delta_{16}^3,\delta_{16}^4\} \in F$ is a cycle of size 4. Further calculations can yield the basin of this cycle as $\{\delta_{16}^k \mid k \in [1,8] \cup [13,16]\}$. Finally, converting the results into binary form [13], we obtain the attractors as {0111} and {1111,1110,1101,1100}, the basins of which are {0111, 1000, 1001, 0000, 1010, 1011, 0001, 0010, 0100, 0011,0110,0101} and {1111,1110,1101,1100,1001,1011, 0001,0011,1010,1000,0010,0000}, respectively. The results coincide with the state transfer graphs in Figs. 3 and 4.

Next, we calculate the basin entropies. Figure 5 is the entire state transfer graph of the system, which shows clearly the overlapping and nonoverlapping parts of the basins. We first use the rigorous method to calculate $\alpha_{\rho_1\xi}$ and $\alpha_{\rho_2\xi}$, and we obtain

$$\alpha_{\rho_1\xi} = 0.55, \quad \alpha_{\rho_2\xi} = 0.45.$$

Hence, the weights of the fixed point and the cycle are

$$\omega_{\rho_1} = \sum_{\xi} \alpha_{\rho_1\xi}\beta_\xi + \lambda_{\rho_1} = 0.525,$$
$$\omega_{\rho_2} = \sum_{\xi} \alpha_{\rho_2\xi}\beta_\xi + \lambda_{\rho_2} = 0.475,$$

respectively. So the basin entropy of the network is

$$h = -\omega_{\rho_1}\ln\omega_{\rho_1} - \omega_{\rho_2}\ln\omega_{\rho_2} = 0.6919.$$

We compare this result with that obtained by the approximate method based on an even partition of the overlapping part of the basins. In the latter case, the weights of the fixed point and the cycle are given as

$$\omega_{\rho_{1,2}} = 4/16 + 8/(2 \times 16) = 0.5.$$

The basin entropy of the network then takes a value of $h' = 0.6931$. Observe that the error $\Delta h = 0.0012$ is very small. This may be indicative of a certain effectiveness of the approximate method.

VIII. CONCLUDING REMARKS

We have presented an approach to study ARBNs based on the STP technique developed recently in Ref. [13], which recasts the logical dynamics of ARBNs into a standard linear iterative dynamics and hence allows for analysis of ARBNs in terms of matrix algebra. Our approach gives in principle a resolution of finding all attractors and their basins in ARBNs with arbitrary connectivity. We also discussed the phenomenon of overlapping basins that occur only in ARBNs, and we proposed two different methods for calculating the basin entropy of ARBNs; one is rigorous but computationally costly and the other approximative but very simple.

Here, we would like to note that the approach proposed in a recent paper [7] can only numerically calculate the weights of basins in ARBNs without determining the basins themselves. In contrast, our approach can give an exact algebraic characterization of all attractors and their basins for ARBNs, and it involves mainly matrix computations in application. This may give rise to difficulties in calculation as the dimension of the state transition matrix grows exponentially ($2^N \times 2^N$) with the system size (N). Hence, developing effective algorithms or approximate techniques for the present approach will be a challenging problem in future work.

ACKNOWLEDGMENTS

This work was supported by National Basic Research Program of China (973 Program, No. 2012CB821200) and NSFC (No. 60974064).

[1] S. Kauffman, J. Theor. Biol. **22**, 437 (1969).
[2] S. Kauffman, C. Peterson, B. Samuelsson, and C. Troein, Proc. Natl. Acad. Sci. USA **100**, 14796 (2003).
[3] B. Huberman and N. Glance, Proc. Natl. Acad. Sci. USA **90**, 7716 (1993).
[4] K. Klemm and S. Bornholdt, Phys. Rev. E **72**, 055101 (2005).
[5] J. Aracena, E. Goles, A. Moreira, and L. Salinas, Biosystems **97**, 1 (2009).
[6] F. Greil and B. Drossel, Phys. Rev. Lett. **95**, 048701 (2005).
[7] A. Shreim, A. Berdahl, F. Greil, J. Davidsen, and M. Paczuski, Phys. Rev. E **82**, 035102 (2010).

[8] J. E. S. Socolar and S. A. Kauffman, Phys. Rev. Lett. **90**, 068702 (2003).

[9] B. Samuelsson and C. Troein, Phys. Rev. Lett. **90**, 098701 (2003).

[10] B. Drossel, T. Mihaljev, and F. Greil, Phys. Rev. Lett. **94**, 088701 (2005).

[11] B. Drossel, Phys. Rev. E **72**, 016110 (2005).

[12] V. Kaufman, T. Mihaljev, and B. Drossel, Phys. Rev. E **72**, 046124 (2005).

[13] D. Cheng, H. Qi, and Z. Li, *Analysis and Control of Boolean Networks* (Springer, London, 2011).

[14] P. Krawitz and I. Shmulevich, Phys. Rev. E **76**, 036115 (2007).

[15] P. Krawitz and I. Shmulevich, Phys. Rev. Lett. **98**, 158701 (2007).