General method to find the attractors of discrete dynamic models of biological systems

Xiao Gan and Réka Albert

Phys. Rev. E **97**, 042308 — Published 17 April 2018

DOI: 10.1103/PhysRevE.97.042308

A general method to find the attractors of discrete dynamic models of biological systems

Xiao Gan
Department of Physics
Penn State University
U.S.A
xxg114@psu.edu

Réka Albert
Department of Physics
Penn State University
U.S.A
rza1@psu.edu

Abstract

Analyzing the long-term behaviors (attractors) of dynamic models of biological networks can provide valuable insight. We propose a general method that can find the attractors of multi-level discrete dynamical systems by extending a method that finds the attractors of a Boolean network model. The previous method is based on finding stable motifs, subgraphs whose nodes' states can stabilize on their own. We extend the framework from binary states to any finite discrete levels by creating a virtual node for each level of a multi-level node, and describing each virtual node with a quasi-Boolean function. We then create an expanded representation of the multi-level network, find multi-level stable motifs and oscillating motifs, and identify attractors by successive network reduction. In this way, we find both fixed point attractors and complex attractors. We implemented an algorithm, which we test and validate on representative synthetic networks and on published multi-level models of biological networks. Despite its primary motivation to analyze biological networks, our motif-based method is general and can be applied to any finite discrete dynamical system.

PACS numbers: 07.05.Tp, 87.15.A, 87.16.Yc, 87.18.Mp, 89.75.Fb

I. INTRODUCTION

Dynamic modeling is a valuable avenue for understanding the emergent properties of interacting biological systems [1, 2]. Networks, with their nodes representing biological entities and their edges representing interactions, can connect the interactions among cellular constituents (e.g. mRNAs, proteins or small molecules) to cell-level functions or behaviors [3, 4]. Once a network is constructed, a dynamic model can be created next. Each node is characterized with a state variable, representing its abundance, concentration or activation level [5]. The state variable will evolve over time according to a regulatory function that depends on the regulators of the node. The state variables and regulatory functions of a dynamic model can be discrete or continuous. Discrete modeling is particularly powerful in biological models in that it can capture the system's behavior without the need for much kinetic detail [6-8]. Such detail, including reaction stoichiometry and kinetic rates, is often difficult to obtain in experiments, and for most systems, especially large networks, the existing knowledge is insufficient to effectively inform continuous models [9]. In this work we focus on discrete dynamic models.

Attractors are long-term behaviors of a dynamic system, and represent system-level outcomes. They are especially important for biological systems because they represent biological phenotypes. For example, in a cell signaling network, attractors can correspond to cell types, cell fates or behaviors, including cyclic behaviors such as circadian rhythms and the cell cycle [10, 11]. Therefore, finding the attractor repertoire of a network model is an important goal. However finding all attractors (including cyclic and complex attractors) is challenging due to the complex dynamics of networks [12]. Thanks to the strong advances in understanding network structure [13-16], a promising way to tackle this problem is to try to find the attractors based on the network topology and the key features of the network's dynamics, instead of from its detailed dynamics [17]. For example, R. Thomas related the conditions of multi-stability and cyclic attractors to positive and negative feedback loops, respectively [18]. Boolean models, which characterize each node with two states and describe regulation in a parameter-free manner, are most strongly based on the network structure. Many methods exist for finding attractors of Boolean networks [19-22]. Although for some systems Boolean modeling is appropriate, often at least a subset of the nodes needs to be characterized by multiple levels, in order to accurately describe experimentally observed relative outcomes in case of combinations of inputs [23, 24]. For example, multiple elements of the signal transduction network that underlies light-induced opening of microscopic pores on plant leaves were observed to have different activity levels under red light, blue light, and white (combined) light [25]. Three levels also allow the separate representation of upregulation or downregulation compared to a baseline/normal level [26]. Current approaches to attractor (mainly fixed point) identification in multi-level models use exhaustive search, model checking methods or polynomial algebra [27-29]. Yet, there is still an unmet need for a general method that can effectively find all attractors (fixed points and complex attractors) of a multi-level model.

In this paper, we propose a general method that can find both fixed points and complex attractors of any finite multi-level model. Our method is an extension of a Boolean attractor finding method proposed by Zañudo & Albert [30]. We test and validate our method on synthetic networks and on a collection of biological models from the literature.

II. METHODS

In this section we give background information on discrete dynamic modeling and attractors, and then an overview of our method. In sub-sections C to I we describe each step of the method in detail.

A. Discrete dynamic modeling and attractors

Discrete dynamic models require minimal parameterization, yet they can capture important biological emergent properties, and are widely used in describing biological networks [12, 31]. These models use discrete time (implemented through update schemes). There are deterministic update schemes such as synchronous update, where all nodes are updated simultaneously at each time step according to their regulatory function [32], or asynchronous schemes with fixed time delays [33]; there are also stochastic update schemes [34], for example a general asynchronous update where in each time step, one node from the network is randomly chosen to update [35]. By considering multiple replicate simulations, general asynchronous update in effect samples all kinds of rates. It is motivated by the fact that the temporal details of biological processes are difficult to obtain and usually insufficiently known. By considering every kind of rates of states transitions, this update method is capable of covering multiple timescales involved in intracellular processes, making up for incomplete knowledge of the reaction timescales in biological network modeling, while synchronous update can lead to spurious behaviors [36]. Therefore it is applied frequently in biological models, in both simulations [37] and theoretical analysis [38, 39].

An attractor can be described as one of the smallest self-contained set of states, i.e. a set of states from which only states in the same set can be reached. Attractors include steady state attractors (fixed points), and complex (oscillating) attractors where a subset of the nodes do not take fixed values. In discrete models, attractors can also be defined as the terminal strongly-connected-components (SCCs) of the state transition graph (STG). An STG of a dynamic model is the graph wherein each node represents a state of the system, and each edge represents a state transition. The nodes in a terminal SCC of the STG are self-contained as they cannot reach nodes other than themselves, and are therefore attractors of the system.

In a discrete dynamical system, the fixed point attractors are independent of the update scheme; on the other hand, complex attractors may depend on the update scheme of the system. This is intuitive, as the edges of a STG can be different for different update schemes. An example is provided in Appendix D1. Since the general asynchronous update allow all kinds of rates and timing, complex attractors found under general asynchronous update are invariant with respect to arbitrary fluctuations in the rates of the processes involved [40]. In this paper, we will focus on general asynchronous update.

An accurate method to find all attractors of discrete models is to perform an exhaustive search in the state space. However this is not practical because the state space of a network scales exponentially with its size. Even for the simplest, Boolean model, the size of the state space of an N-node network is 2^N, which is too large for exhaustive search. There has been a lot of effort to develop methods to find attractors in the Boolean framework [19-22], but there are only a few methods that can find attractors of multi-level models, and they have special constrains when finding complex attractors. For example, Dubrova et al. proposed an SAT-based bounded model checking method that can only find complex attractors in a synchronous update scheme [27]. Hinkelmann et al. converted the attractor finding problem into solving polynomial equations; this method can only find complex attractors of a limited size [28]. Our method does not explicitly consider time and does not enumerate the system's trajectories. Instead, it combines graph topology and regulatory functions into an expanded graph representation. Because this expanded network is much smaller than the size of the state space, our method can work on networks of larger size. Our method is comprehensive in the broad family of dynamical systems wherein one node changes state at any given time instant.

B. Overview of our motif-based attractor identification method

The idea of our method is to translate the attractor identification problem into a graph theoretical problem by creating an expanded representation of the network that incorporates all the regulatory functions, then identifying certain motifs (subgraphs) of this expanded network [41]. We will refer to our method as the motif-based attractor identification method, or 'motif-based method' for short.

We first represent each state of each original node with a Boolean virtual node. The 'ON' state of the virtual node means that the original node is in the state embodied by the virtual node. The regulatory function of a virtual node is a quasi-Boolean function, whose inputs are virtual nodes, expressed in an appropriate disjunctive normal form. This disjunctive normal form is obtained by summing up the input combinations that yield the 'ON' state for the virtual node.

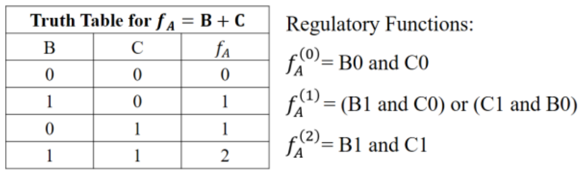
Then an expanded network containing all information expressed in the regulatory functions can be established. The expanded network is obtained from the original one by the following operations: 1. Include each virtual node in the expanded network, and connect all the virtual node's regulators to it; 2. for each 'and' rule in the regulatory functions, create a composite node, and re-wire the edges from the input nodes of the 'and' rule to this composite node, then connect the composite node to the regulated (target) node. The original edges from the input nodes of the 'and' rule to the target node are removed. The expanded network allows one to distinguish between co-pointing interactions that are combinatorial in nature (i.e. they are combined by 'and' rules) from co-pointing interactions that are individually sufficient (i.e. they are combined by 'or' rules).

We use the term "motif" for strongly connected components of the expanded network that satisfy certain properties (which we will describe later). Depending on the virtual nodes involved in the motif, we define stable motifs, which correspond to stabilized states of the constituent nodes, and oscillating motifs, which are candidates for oscillations of the constituent nodes.

After the motifs are found, plugging in the node states specified in the motifs into the regulatory functions of their target nodes will specify the states of these nodes, therefore reducing the network. Then more motifs can be found in the reduced network, and this reduction process can be done iteratively. Ultimately, the motif sequence we find in the iteration process will determine the attractor. In the following sub-sections we describe the details of each step.

C. Quasi-Boolean formalism of multi-level models

We establish a formalism where multi-level regulatory functions become Boolean-like. We treat each level (state) of a multi-level node as a separate node, called a virtual node. For example, if a node A has 3 different levels, 0, 1, and 2, then 3 virtual nodes for A, namely A0, A1, A2, are created in our formalism. Each virtual node is like a Boolean variable, and the combination of all virtual nodes represents the state of the original node. We will refer to these virtual nodes as 'sibling nodes' of each other. For example, original state A=2 (where for simplicity the node state is represented by the node name) will now be represented as the combination A0=0, A1=0, A2=1. Note that one and only one of the virtual nodes takes value 1, while all other virtual nodes, i.e. its sibling nodes, must all be 0. Then we write the regulatory function of each virtual node in a Boolean disjunctive normal form, by treating each input combination as a conjunctive clause and then connecting all conjunctive clauses that yield the same target node level with the Boolean 'or' operator. Figure 1 demonstrates the example of converting the regulatory function f_A = B+C into a set of quasi-Boolean regulatory functions of virtual nodes.



Truth Table for $f_A = \mathrm{B} + \mathrm{C}$		
B	C	f_A
0	0	0
1	0	1
0	1	1
1	1	2

Regulatory Functions:

$f_A^{(0)}$= B0 and C0

$f_A^{(1)}$= (B1 and C0) or (C1 and B0)

$f_A^{(2)}$= B1 and C1

FIG. 1 Demonstration of the construction of a quasi-Boolean regulatory function. A 3-level node A has regulatory function: f_A =B+C, where B and C both have 2 levels. From the truth table, one can identify the regulatory function for each virtual node of A, by connecting all conjunctive clauses that yield the same state of A with the Boolean 'or' operator. In this way, each virtual node's regulatory function will have a Boolean disjunctive normal form.

Note that the Boolean 'not' rule is absent from this formalism, because we have assigned virtual nodes to all states of nodes. Negation is now replaced with activation by the sibling nodes. We will proceed through the rest of our analysis based on the regulatory functions of the virtual nodes, instead of the functions of the original nodes.

We require the regulatory functions to be written in a disjunctive normal form with all of their prime implicants present, or in other words, in the Blake canonical form [42]. A minterm is a combination of inputs that yields the value 1 for a Boolean expression. An implicant is a 'covering' (sum or product) of minterms in a Boolean function; a prime implicant of a function is an implicant that cannot be covered by a more general (more reduced) implicant. For example, the Blake canonical form of the regulatory function 'f_A = B *and* C or D *and not* C' is 'f_A = B *and* C or D *and not* C *or* B *and* D', as the conjunctive clause 'B *and* D' is also a prime implicant of A. This form is not preferred in Boolean models because of its redundancy, but it is necessary for the creation of the expanded network, because it explicitly contains all sufficient conditions to activate a virtual node. The Quine-McCluskey (QM) algorithm finds the Blake canonical form of a Boolean function. We extend this algorithm to multi-level models.

D. Multi-level Quine-McCluskey algorithm

To obtain the Blake canonical form of a multi-level function, we developed a multi-level version of the QM algorithm. The original QM algorithm not only finds all prime implicants but also minimizes the function [43-45]. We aim to find all prime implicants and omit the latter step.

The idea of the QM algorithm is that, if multiple minterms cover all states of a node, these minterms can be merged and the node can be eliminated from the function. For example, in a Boolean case, A *and* B *or* A *and not* B = A. Similarly, if all states of a node in a multi-level function are covered by certain minterms, these minterms can be merged. For example, if B has 3 states, then A1 *and* B0 *or* A1 *and* B1 *or* A1 *and* B2 =A1. The key property here is B0 *or* B1 *or* B2 =1; or in general, $\mathrm{N}^{(0)}$ *or* $\mathrm{N}^{(1)}$ *or* $\mathrm{N}^{(2)}$ *or* … *or* $\mathrm{N}^{(m-1)}$=1, where m is the number of states of node N has and $N^{(i-1)}$ represents the ith state of N. We call this the completeness condition. The main difference of the multi-level functions compared to a Boolean function is that the completeness condition becomes implicit. There is also a uniqueness condition, which can be written as $\mathrm{N}^{(i)}$ *and* $\mathrm{N}^{(j)} = 0, \forall\, i \neq j$. The interpretation is that N can only take a single state. Together the completeness and uniqueness conditions mean that at any given time node N can take one and only one state from its possible states, which is a natural requirement. These conditions are true in the Boolean formalism (A *or not* A = 1, A *and not* A =0). However, in the multi-level formalism where we represent each node state separately, we will need to separately impose these two conditions. Specifically, the multi-level QM requires the completeness condition to merge minterms.

The systematic merging can be done in a way demonstrated in Figure 2. Suppose a virtual node state D1 has its regulatory function expressed in truth table format. One can then re-arrange the minterms into groups, based on how many zeros each minterm has. Then one can start merging by checking minterms in neighboring groups that are different by one node. If the minterms cover all states of that node, then they can be merged. In the example demonstrated in Figure 2, m1 (002), m5 (012) and m6 (022) differ in the state of node B, and these three minterms cover all possible states of B, so we can merge them to get '0X2' in the 1st row on the right, as a merged term. This process is done repeatedly until all minterms are considered. Any leftover minterms that did not get merged are prime implicants, e.g. (011) in Figure 2. The merged terms will contain 'X's representing merged nodes. Next, one treats the 1st order merged table in the same way, i.e. re-arrange according to the number of zeros, and try to merge into a 2nd order merged table. The difference is that 'X's are treated as a separate state of the variable that cannot be merged. For example, (X01) and (X11) are different by 1 node and may be considered as candidates for merging, while (X01) and (0X1) are different by 2 nodes and cannot be merged. This process is done iteratively until no more merging can be done. All ‘leftover’ terms are prime implicants. In Figure 2, nothing can be merged after 1st order, so we get a final prime implicant form of D1 as $f_D^{(1)}$ = A0 *and* B1 *and* C1 *or* A0 *and* C2 *or* B0 *and* C2 *or* A1 *and* C0 *or* B2 *and* C2 *or* B2 *and* C0. We discuss the performance of the algorithm in Appendix A, and a description of the implementation is provided in Appendix B.

A	B	C	$f_D^{(1)}$
0	0	2	1
0	1	1	1
0	1	2	1
0	2	0	1
0	2	2	1
1	0	0	1
1	0	2	1
1	1	0	1
1	2	0	1
1	2	2	1

Minterm	A	B	C
m1	0	0	2
m2	1	0	0
m3	0	2	0
m4	0	1	1
m5	0	1	2
m6	0	2	2
m7	1	0	2
m8	1	1	0
m9	1	2	0
m10	1	2	2

A	B	C
0	X	2
X	0	2
1	X	0
X	2	2
X	2	0

+

A	B	C
0	1	1

FIG. 2 Example of the multi-level Quine-McCluskey algorithm. A Boolean node D is regulated by a Boolean node A and two 3-state nodes B and C. The original function of D is shown in a truth table on the top left, in a form summarizing all input combinations that yield $f_D^{(1)}$ =1. The top right table shows the minterms sorted according to the number of zeros in them. From this table, one can merge the terms between layers that are different by 1 digit, if all states of the difference node are present within the two layers. The result of the merging is shown below. Merged terms are represented by an ‘X’. There are 5 leftover terms after 1st order merging, and there is 1 leftover term after 0th order merging. The sum of all six terms is the final expression.

E. The expanded network representation

After all functions are transformed into the proper form, we create an expanded network, which is a representation of the network with regulatory functions embedded. The expanded network is obtained from the original network by applying the following operations: 1. Include each virtual node in the expanded network, and connect its regulators to it; 2. for each ‘and’ rule in the regulatory functions, create a composite node, and re-wire the edges from the input nodes of the ‘and’ rule to this composite node, then connect the composite node to the regulated node. The original edges from input nodes of the ‘and’ rule to the target node are removed. Figure 3 exemplifies the construction of an expanded network from a regulatory function. To construct the entire expanded network, all virtual nodes and all interactions must be created.

Regulatory function:

$f_A^{(0)}$= B0 or (C1 and B1)

$f_A^{(1)}$= B1 and C0

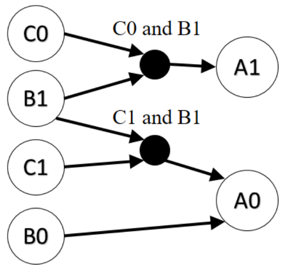


FIG. 3 Construction of an expanded network from a regulatory function. Virtual node A0 has function $f_A^{(0)}$ = B0 or (C1 and B1), so in the expanded network, B0 is connected directly to A0; C1 and B1 are connected indirectly to A0 via composite node 'C1 and B1'. A1 has function $f_A^{(1)}$ = C0 and B1, so C0 and B1 are connected indirectly to A0 via composite node 'C0 and B1'.

The expanded network contains not only the network structure, but also all information about the regulatory functions. Furthermore, interactions of a combinatorial nature are separated, as all ‘and’ rules have become explicit nodes. In this way, the expanded network makes it easy to identify a sufficient condition to activate a node: a virtual node will have state 1 if any of its regulator virtual nodes is 1, or if any of its regulators that is a composite node has all its input virtual nodes being 1, regardless of the states of the rest of its regulators. Following this intuition, a cycle in the expanded network that satisfies the above criterion will be self-sufficient to stabilize.

This leads to the definition of stable motifs.

F. Stable motifs

A stable motif is a subgraph of the expanded network that can stabilize on its own. We define it in the following way: a stable motif is a strongly-connected-component (SCC) in the expanded network that satisfies: (1) the SCC contains no sibling node pairs; (2) if the SCC contains a composite node, all of its input nodes must also be in the SCC. The first condition is a natural requirement for a stabilized state of the original node; the second condition is about the nature of the Boolean ‘and’ operator, as all inputs must be present to activate the ‘and’ function. In our algorithm we identify stable motifs as the smallest SCCs that satisfy the above conditions. Figure 4 shows the expanded network and stable motifs of a three node network.

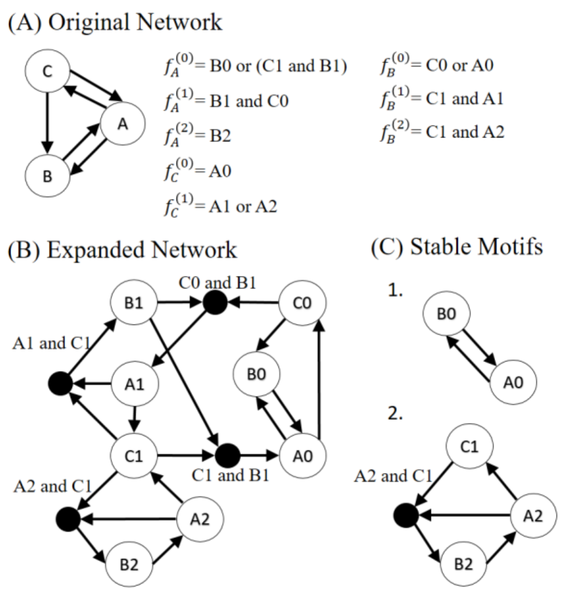


FIG. 4 Illustration of stable motif identification in a three-node network. (A) The original network and the regulatory functions of each node; (B) The expanded network is constructed according to the steps in section I.E, and then the stable motifs are found by their definition in I.F. (C) Stable motifs found in this example. The first stable motif, A0, B0, corresponds to a fixed point attractor of the system A=0, B=0, C=0. The state C=0 is found by plugging A=B=0 into the regulatory function of C. The 2nd stable motif corresponds to another fixed point attractor A=2, B=2, C=0.

In order for stable motifs to be correctly recognized, the regulatory functions must contain all prime implicants. If a prime implicant is missing, a sufficient condition for a node to stabilize is missing, which would lead to incorrect identification of stable motifs. This is why we require the Blake canonical form of regulatory functions.

There is a one-to-one correspondence between a stable motif and a partial fixed point of the system (which is defined as a state in which a subset of nodes stabilize regardless of the state of the rest of the system). The proof of this statement is provided in Appendix C. Consequently, by finding all stable motifs we find all fixed points or partial fixed points of the system.

G. Oscillating motifs

An oscillating motif is defined as the largest SCC in the expanded network that satisfies: (1) at least one virtual node in the SCC has at least one sibling node in the SCC; (2) if the SCC contains a composite node, all its input nodes must also be in the SCC. In contrast to nodes in stable motifs, an oscillating node must be able to enter at least two states, so the first condition is necessary. The second condition is also necessary due to the combinatorial nature of the composite node.

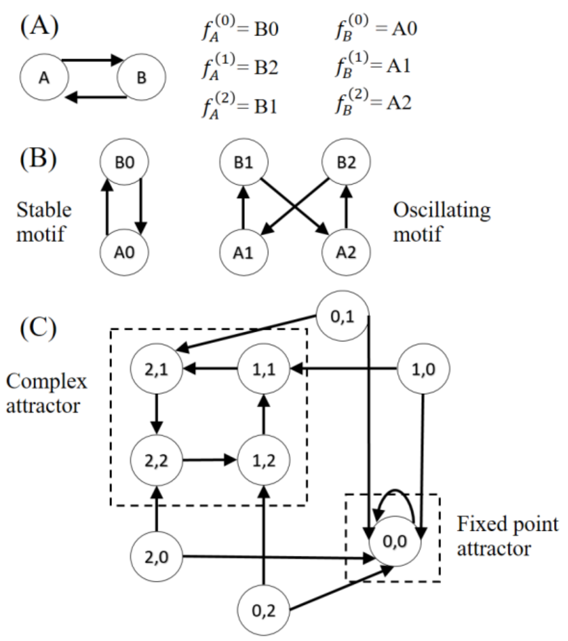


FIG. 5 An example of an oscillating motif in a multi-level network. Panel (A) shows the network and regulatory functions; panel (B) indicates the expanded network and motifs. A0 and B0 form a stable motif, indicating a fixed point A=0, B=0; while A1, A2, B1 and B2 form an oscillating motif, indicating a possible complex attractor involving states A=1, A=2, B=1 and B=2. Panel (C) indicates the state transition graph of the system when using general asynchronous update. The stable motif and oscillating motif identified in 5B correspond to a fixed point and a complex attractor, respectively.

Unlike the relation between stable motifs and partial fixed points, there is no one-to-one correspondence between oscillating motifs and complex attractors, because complex attractors are dependent on the timing of individual events [37] (see Appendix D1 for an example). Our motif-based method is based on network structure and regulatory functions and is independent of timing, thus it cannot find timing-dependent complex attractors. General asynchronous update prunes timing-dependent complex attractors in discrete framework, and all complex attractors under this update are proven to be based on negative feedback loops [38, 39]. These complex attractor are also reliable under perturbation, in contrary to timing-dependent complex attractors [40]. Therefore the complex attractors identified by our method should be consistent with the complex attractors under general asynchronous update. We propose that for every complex attractor of the discrete dynamic system under general asynchronous update, there is a set of oscillating motifs and their downstream that contain the virtual nodes representing all the states visited by the oscillating nodes. We sketch the proof of this proposition in Appendix C. In our benchmarks presented in section III. B., this proposition was never violated. Figure 5 shows an example of a complex attractor in a multi-level network model. This example also illustrates the coexistence of a fixed point attractor and a complex attractor for different states of the same nodes (See Appendix D4 for more detail).

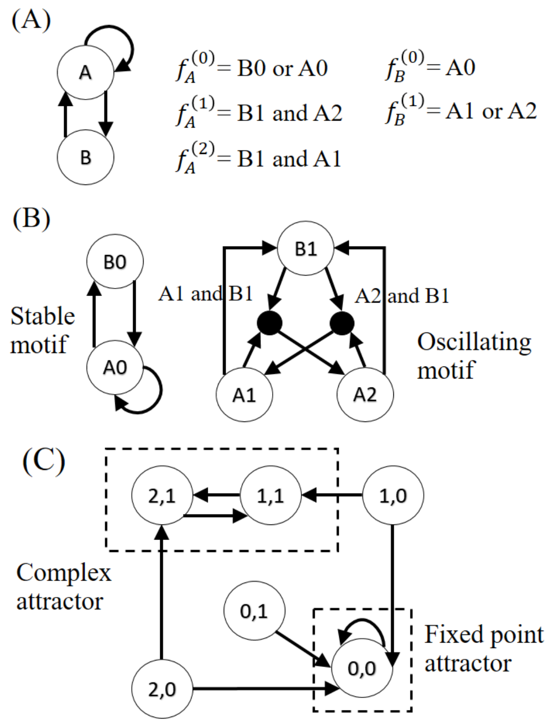


FIG. 6 An example of an oscillating motif that contains a stabilized node. (A) The network and regulatory functions. (B) The expanded network and motifs. The oscillating motif contains only one virtual node of B, meaning that B will stabilize at 1 in the complex attractor. (C) The state transition graph using general asynchronous update. There are two attractors: a fixed point attractor, and a complex attractor.

There is a difference between the criteria of oscillating motifs in the Boolean and multi-level case: in the Boolean case, all nodes in an oscillating motif must oscillate [30], while in the multi-level case, an oscillating motif can allow stabilized nodes. An example of a complex attractor corresponding to an oscillating motif with a stabilized node is shown in Figure 6. We illustrate several additional properties of oscillating motifs in Appendix D.

H. Iterative motif reduction yields the attractors of the system

The source (unregulated) nodes of a network that stabilize in a fixed state can be reduced prior to any attractor identification process. The corresponding fixed states can be substituted into the regulatory functions of the nodes they regulate. This can be done iteratively until no source nodes are present in the network, without affecting the attractor repertoire of the system [46, 47]. For some biological networks, this reduction alone can reduce a large fraction of the network model, leading to a much simplified model.

Once motifs are identified, we can plug in the states of the nodes specified in the motifs into the expanded network, as if these nodes were source nodes, to further reduce the network. For stabilized nodes, the stabilized virtual node takes value 1 and its sibling nodes are set to 0; for oscillating nodes, their corresponding virtual nodes are marked as oscillating, and their sibling nodes excluded from the oscillating motif are set to 0. Certain nodes downstream of the motifs may stabilize as a result. In this way, a reduced version of the network model is obtained. We then identify stable motifs and oscillating motifs in the reduced network and substitute the corresponding virtual node values again, until this cannot be done any more. By the end of this process all nodes will either become a part of a motif, or be downstream of a motif and be determined by that motif, and we will have obtained a set of motif sequences. If no oscillating motifs are found in a motif sequence and at the end of the process the network has reduced completely, all nodes will be stabilized, and we will have obtained a fixed point attractor. If oscillating motifs are found in a motif sequence, at the end of the process we will find some (possibly none) of the nodes stabilized, while some other nodes oscillating. We call this result a quasi-attractor. Specifically, a quasi-attractor will indicate the unique state of each stabilized node and it will give a set of states among which a potentially oscillating node oscillates. This quasi-attractor is likely (but not guaranteed) to correspond to a complex attractor (see Figure 10 in Appendix D for example). Under general asynchronous update, since all partial fixed points correspond to stable motifs, and all complex attractors correspond to oscillating motifs, all attractors will be covered with our motif-based method. Note that there is no exact match between the actual number of complex attractors and the number of quasi-attractors found by our method (see Appendix C and D for proof and examples).

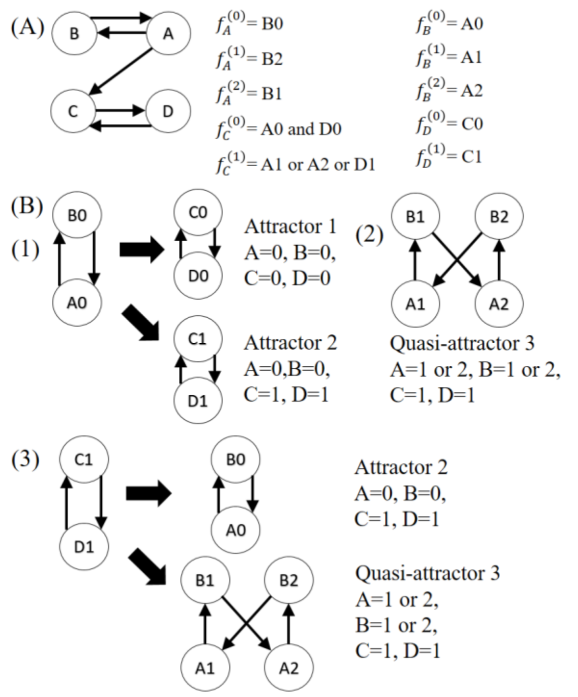


FIG. 7 Attractor identification for a four-node network by a motif succession diagram. A. The network and the regulatory function of each node. B. Motif succession diagram. Three motifs are found from the original network, including 2 stable motifs (A0, B0), (C1, D1), and one oscillating motif (A1, A2, B1, B2). For each motif, the values of the nodes in the motif are plugged into the regulatory functions, reducing the network. Then new motifs are identified from the reduced networks. The sequences corresponding to the three motifs are labeled (1), (2) and (3).

The reduction process can be represented as a motif succesion diagram, which is the diagram of the motifs obtained successively in the iterative network reduction process [48]. Figure 7 illustrates a motif succession diagram, where iterative network reduction based on identified motifs leads to the identification of attractors and quasi-attractors. The original network has two stable motifs (A0, B0), (C1, D1), and one oscillating motif (A1, A2, B1, B2). When the stable motif (A0, B0) is chosen, the network is reduced down to two nodes, C and D, with new regulatory functions $f_C^{(0)}$ = D0, $f_C^{(1)}$= D1, $f_D^{(0)}$ =C0, $f_D^{(1)}$=C1. Two new stable motifs, (C0, D0) and (C1, D1) are found in the reduced network, leading to two attractors Attractor 1: A=0, B=0, C=0, D=0 and Attractor 2: A=0, B=0, C=1 D=1. When the oscillating motif is chosen, A0 and B0 become 0, and as a consequence C0 and D0 become 0, thus C=D=1. The system is thus in a quasi-attractor in which A and B oscillate between 1 and 2 and C=D=1. When the stable motif (C1, D1) is chosen, the regulatory functions of A and B stay the same, thus either the (A0, B0) stable motif or the oscillating motif can come next. Both yield already encountered (quasi)-attractors (see Figure 6). Thus Attractor 1 is reached if stabilization of (A0, B0) is followed by (C0, D0), Attractor 2 is reached in case of stabilization of (A0, B0) and (C1, D1), in either order, and quasi-attractor 3 is reached due to the oscillating motif. In general, a (partially) ordered sequence of motifs determines a fixed point attractor or quasi-attractor, similarly to the Boolean case [48].

I. Description of the motif-based algorithm

Here we summarize the steps of the implementation of the motif-based algorithm [1]. The algorithm takes as input a set of regulatory functions and specific values for each source node. For a source node A whose value is uncertain, one can define its regulatory function as itself, i.e. $f_A = A$. In this way each virtual node that corresponds to A will have a self-loop, which is also a stable motif. Thus all possible values of A are considered.

1. Reduce the source nodes of the network model by plugging their values into the regulatory functions of the nodes they regulate. Repeat until no source node is present.
2. Transform the regulatory functions to Blake canonical form using the multi-level Quine-McCluskey algorithm.
3. Create the expanded network according to the definition in section I.E.
4. Search the expanded network for stable motifs and oscillating motifs.
5. For each stable motif and oscillating motif identified, create a copy of the network, with the node states specified in the motif plugged into the regulatory functions of their targets. In the case of oscillating motifs, the virtual nodes in the oscillating motif are marked, and their sibling nodes that are not in the motif are set to 0. In addition, for each oscillating motif, create a copy of the network with all virtual nodes downstream of the oscillating motif marked.
6. Repeat 1, 2, 3, 4, and 5 until no more motifs can be identified. In step 1, the reduction process, virtual nodes marked as potentially

[1] The source code is available on GitHub: https://github.com/jackxiaogan/Multi-level_motif_algorithm.

oscillatory are not reduced when evaluating regulatory functions.

7. Discard duplicate attractors.

The final result of the algorithm will be a set of attractors or quasi-attractors. Each of these (quasi) attractors will indicate a state (or multiple possible states) for each node. For each stabilized node, its unique stabilized state is given; for a potentially oscillating node, the multiple states among which it potentially oscillates are given.

III. RESULTS

To test the effectiveness of our motif-based attractor identification method, we apply it to an ensemble of synthetic networks and biological networks from the literature.

A. Benchmark on synthetic networks

We test the motif-based algorithm on synthetic networks of different size, ranging from 10 to 40. To approximate biological networks, we first generate networks where the in-degree is k=2 for each node and the network is otherwise random [49, 50]. Next, we generate the number of states for each node. For multi-level ensembles, we generate number of states according to an equal probability of having 2 or 3 states. For Boolean ensembles all nodes have 2 states. Then we randomly generate a regulatory function among those consistent with the number of regulators and number of states for each node. The generation process of regulatory functions is described in Appendix E.

To test whether the motif-based algorithm finds attractors correctly, we perform simulations similar to Wang et al. [51] and Zañudo et al. [30]. We start from different random initial conditions, and let the system evolve for T_{step} effective time steps. We used general asynchronous update, where at each time step, one node is randomly chosen and its state is updated according to its regulatory function. If the new state of the node is the same as before, another node will be selected within the same time step, until the selected node changes state. If no node can reach a new state, a fixed point attractor is reached. If no fixed point attractor is reached within T_{step} effective time steps, we evaluate whether the system is in a complex attractor by determining the corresponding partial state transition graph (STG). Note that this sampling method is heuristic, and is likely to miss attractors when the state space is large. For each fixed point attractor found by simulation, we check whether it is predicted by our motif-based algorithm. In addition, for each predicted fixed point or partial fixed point we check whether there is a simulated attractor that contains the same stabilized nodes in the same states. If a pair of predicted and simulated fixed points passes both checks, we categorize them as identical. If a predicted partial fixed point passes the second check, we call it consistent with the simulated attractor. Complex attractors depend on the update scheme (i.e. on the timing), so there cannot be a definitive conclusion. The expectation (based on our proposition presented in II.G) is that the set of nodes found to oscillate in a simulation should be a subset of the nodes predicted to oscillate by our motif-based algorithm. If this is indeed the case (in addition to the stabilized nodes, i.e. the partial fixed points, being consistent), we say that the attractors are highly consistent. In all tests, we found identical fixed points and highly consistent complex attractors with the sampling method. The runtime of the motif-based algorithm increases exponentially with the number of nodes, and increases faster on the ensemble of multi-level networks than on an ensemble of Boolean networks, as expected (Table I). From the table, the motif-based algorithm would not be practical for large networks with more than 50 nodes or too many multi-level nodes. The important question is whether the algorithm is practical for biological network models existing at present or constructed in the near future. To estimate the answer to this question, we test our algorithm on published multi-level biological network models.

Multi-level Networks				
Size of network	10	15	20	25
Time (s)	0.07	1.1	48	251
Boolean Networks				
Size of network	10	20	30	40
Time (s)	0.07	0.89	74	600

Table I. Benchmark runtime of the motif-based algorithm on synthetic networks of different sizes (number of nodes). For each size, 50-100 random networks with in-degree k=2 are generated. For multi-level networks, each node has 50% chance of having 2 levels and has 50% chance of having 3 levels. In all runs, the attractors found by the algorithm are identical or highly consistent with the attractors found with the sampling method.

B. Tests on biological networks from the literature

The tested models include a signal transduction network model describing stomatal opening in plants [25] whose attractor repertoire we explored before [52]. We also selected 18 models from the model repository of the software tool GINsim, which simulates discrete dynamic models of gene regulatory networks [29]. These 19 models have sizes ranging from 4 to 72 nodes, with 6%-100% of these nodes being multi-level. We run our motif-based algorithm on each model, and compare the results with the results found by GINsim.

To apply the motif-based algorithm, we first convert the GINsim model into a '.txt' file, with regulatory functions suitable for our algorithm[2]. In the few cases where the GINsim framework and our framework are different, we adapt the model to our framework. For example, GINsim allows an 'empty function': '$f_A^{(0)}$ = B0, $f_A^{(2)}$=B1, $f_A^{(1)}$ is empty, i.e. A1 has no function', which our method doesn't allow. In this GINsim example, 'A1' will be visited transiently when node A changes from A0 to A2. We discard the state 'A1'. We can do so because such transient states are never part of an attractor. We also reduce some of the large models before applying our algorithm. The reduction consists of three methods: removing output nodes (nodes with no outgoing edges), removing simple mediator nodes (nodes with one incoming edges and one outgoing edge), and replacing input trees (acyclic sub-networks that contain a source node) with a single source node. These reductions are known to conserve the attractors of the model [46, 47]. In cases where there are a lot of different signal (source node) state combinations, it is not practical to compare all the fixed points found. Instead, we select representative signal combinations corresponding to different biological phenotypes (some of which are indicated as pre-made selections in GINsim), or signal combinations that result in different attractors.

We compare the attractor analysis results by first checking whether the fixed points are identical, and then checking whether the complex attractors are consistent. We find that the fixed points found by the two algorithms are identical, as expected. For complex attractors, it is difficult to get a definite conclusion. GINsim cannot predict complex attractors; it can only simulate the state transition graph (STG) or hierarchical transition graph (HTG) and find the strongly-connected-component from the STG/HTG [53]. The complexity of this method goes up quickly with the increase of the model size. Our method can only predict quasi-attractors, which may or may not be actual complex attractors. Therefore it is impossible to know the complex attractors exactly unless an exhaustive (partial) state space search is performed. If the model is simple enough for GINsim to construct an STG, we check whether the complex attractors found from the STGs are covered by the candidates predicted by our algorithm. We found consistent complex attractor results from the two algorithms: all complex attractors found in simulations are covered by predicted quasi-attractors. The detailed results can be found in the Supplementary File S1 [54].

We also compared the runtime of the two algorithms. For the motif-based algorithm, we record the runtime for each signal combination, then average them. GINsim does not show the actual time spent in computation, so we only record whether the computation completed, and give an estimated time. Note that both algorithms are guaranteed to find solutions given enough computational power, so cases of not completed calculations are due to limited computational resources. All GINsim fixed point computations are done in seconds. The only model wherein the motif-based algorithm did not finish computing had a 72-node strongly connected network. A summary of the results is shown in Table II. The details of the runtime of each model can be found in Supplementary File S1[54].

Network count	Network size	Computational Time	
		Motif algorithm	GINsim STG/HTG
9	4~15	0~8s	0~10s
9	17~36	0s~1h	DNC
1	72	DNC	DNC

Table II. Summary of the runtime of the two algorithms. The networks fall into three categories. The first column is the number of networks in each category. The second column is the range of the network sizes in each category. The 3rd and 4th columns indicate whether motif analysis and GINsim STG/HTG generation was successfully completed or not. For completed analysis, the range of computational time is shown in the table. Otherwise, we indicate DNC (meaning "did not complete"), which includes cases that ran out of memory or did not finish in 6 hours. All tests were run on a personal computer. There is no model where GINsim succeeds and the motif-based algorithm fails. The motif algorithm is successful in 18 of 19 models, while GINsim STG/HTG only works in the small networks of the first category.

[2] The converted models are uploaded to the 'models' folder in: https://github.com/jackxiaogan/Multi-level_motif_algorithm/.

IV. DISCUSSION

Our motif-based attractor identification method connects the structure, regulatory logic and attractors of discrete dynamical systems. The expanded network representation is conceptually similar to Petri nets (as the composite nodes share certain properties with the Petri nets' transition nodes) [55] [56] and also to logic hypergraphs [57] (which represent the group of edges incident on a composite node with a hyper-edge). The innovation of our analysis of the expanded network lies in interpreting the patterns formed by multiple connected regulatory functions. The motifs identified in our expanded network have a strong correspondence with the long-term dynamic behaviors of the modeled system. The expanded network is therefore a good complementary technique to the existing family of techniques to predict the attractor repertoire of discrete dynamical systems.

Our method captures not only fixed points, but also complex attractors. The fixed points of a dynamic system are independent of timing, and will be found accurately. Complex attractors may be timing-dependent. Since our method is based on the structure and regulatory logic of the system, it will capture timing-independent, negative feedback-driven complex attractors. Our method can find all attractors of systems updated by general asynchronous update; for systems updated using other update schemes (i.e. when there exists at least some node synchrony), our method can accurately find fixed points and timing-independent complex attractors, but there may be timing-dependent attractors that our method cannot capture.

The complexity of the motif-based algorithm mainly comes from the identification of cycles. Both stable and oscillating motifs are formed as unions of simple cycles in the expanded network. Identifying simple cycles in a directed graph is known to be NP-complete, with time complexity $O\big((N+E)(c+1)\big)$ using Johnson's algorithm [58], where N is the number of nodes, E is the number of edges, and c is the number of directed cycles. The last can grow faster than 2^N for dense networks. In addition, the introduction of multi-level nodes dramatically increases the number of nodes, especially the number of composite nodes in the expanded network. These facts limit the effectiveness of the motif-based algorithm on networks with a large size, a high number of levels, or with high connectivity. Typical biological network models have a low average degree, around two, and a low number of states for each node (two or three). In addition, only a relatively small fraction of the nodes are in SCCs; i.e. biological networks are not feedback-dense. As we have demonstrated in section III.B, our motif-based method can be successfully applied to these networks. For other types of networks, although our method can theoretically work, the computational complexity may be a challenge. Possible further work on this project include optimizations of the algorithm so it can work on more complex network models, and finding more necessary conditions of multi-level complex attractors to reduce the number of quasi-attractors. A possible way to optimize the algorithm is to add a step to divide the network into SCCs before trying to analyze for motifs, as all motifs can only be found within an SCC. This may dramatically reduce cycle-finding time in networks with SCC 'communities', which is quite common in biological networks.

Although the idea is the same, there are significant differences between the Boolean stable motifs method and our multi-level motif-based method. The most important difference is in the criteria for oscillating motifs, as mentioned in Section II G: the Boolean oscillating motif requires the participation of two (i.e., both) sibling virtual nodes for every node of the motif, while the multi-level oscillating motif does not require that two or more sibling virtual nodes participate for every original node (see the multi-level example in Figure 6). In addition, in the Boolean framework, a fixed point and a complex attractor cannot co-exist for different states of the same node; while in the multi-level case this is possible (see the example in Figure 5 and 6). These differences bring fundamental differences and complications to the design of the algorithm, because in the iterative reduction process toward attractor identification, the Boolean method needs only knowledge of the stable motifs, while the multi-level case needs both stable motifs and oscillating motifs.

The integration of the network structure and regulatory logic in the expanded network can reveal the connectivity patterns that underlie the system's functional repertoire. There can be multiple extensions to this work. For example, in the Boolean case, elementary signaling mode (ESM) has been defined from the expanded network as the minimal set of nodes that can perform signal transduction independently [59, 60]. It can be extended to the multi-level as well to help understand signal transduction a multi-level expanded network.

Another direction is to extend the concepts of expanded network and stable motifs to a continuous framework. If one can distill the causal relationships wherein a certain value of a continuous variable is sufficient to maintain a certain value of another continuous variable, one can construct an expanded network from these relationships, and obtain insight into the system's dynamic repertoire [61].

Furthermore, one can develop the control capability of multi-level motifs. Network controllability has multiple definitions and frameworks to address it [62-65]. Motifs can be used to control the system by driving it into one of its natural attractors. Zañudo et al. proved that in the Boolean case a sequence of stable motifs uniquely determines an attractor, which means that driving certain nodes into their states in a stable motif can drive the network into the corresponding attractor; they also implemented an algorithm to identify driver nodes from Boolean stable motifs [48]. The same principle applies to multi-level stable motifs as well, and the algorithm to find the driver nodes to be controlled can be adapted as well. This is particularly valuable in biological networks, as the control of stable motifs can suggest possible practical interventions to switch the system from an undesired attractor to a desired one. Another possible aspect of control is target control, i.e., driving a single node or small set of nodes into a desired state. This can be done by exploiting more of the sufficiency conditions revealed in an expanded network [66].

V. CONCLUSION

In this paper, we propose a motif-based reduction method to find both fixed points and complex attractors of a discrete dynamic model, by extending an existing method from Boolean to any discrete level. We establish a multi-level formalism and identify motifs from an expanded representation of the multi-level network. Then we iteratively reduce the network according to the motifs to obtain the attractors. Our method is general enough to work on any discrete dynamic model. We demonstrate the method's correctness and effectiveness by implementing an algorithm, and then benchmarking it on synthetic networks, and applying it to biological networks in the literature. In addition, the identification of stable and oscillating motifs offers a way toward attractor control of the network.

VI. ACKNOWLEDGEMENTS

This project was supported by NSF grants PHY 1205840, IIS 1161007, and PHY 1545832. The authors thank Dr. Jorge G. T. Zañudo, Dr. Gang Yang, Prof. István Albert, Nianyuan Bao and Jordan Rozum for helpful discussions.

VII. APPENDIX

Appendix A. Runtime performance of the multi-level Quine-McCluskey algorithm

The computational complexity of the Boolean Quine–McCluskey algorithm grows exponentially with the number of variables, because the problem it solves is NP-hard, and it is shown that the upper bound on the number of prime implicants of a Boolean function with n variables is $3^n \ln(n)$ [67]. Since a Boolean function is a special case of a discrete function , it is straightforward that finding all prime implicants of a multi-level function is at least as complex as finding all prime implicants of a Boolean function. To test whether the multi-level QM algorithm is capable of analyzing biological network models, we benchmark how long it takes for the algorithm to transform all node functions on 100 randomly generated heterogeneous networks. The networks have 50 nodes and have a power law in-degree distribution with exponent -3 and maximum degree 8. Each node has 60% chance of having 2 states, 25% chance of having 3 states, 10% chance of having 4 states, and 5% chance of having 5 states. These parameters exceed the complexity of current multi-level biological models. The result is shown in Figure 8: the multi-level QM algorithm can effectively transform the functions. In addition, we found that within the algorithm, the complexity of identifying stable or oscillating motifs is much more than that of the QM transformation. So we conclude that the complexity of the QM algorithm is acceptable for practical problems.

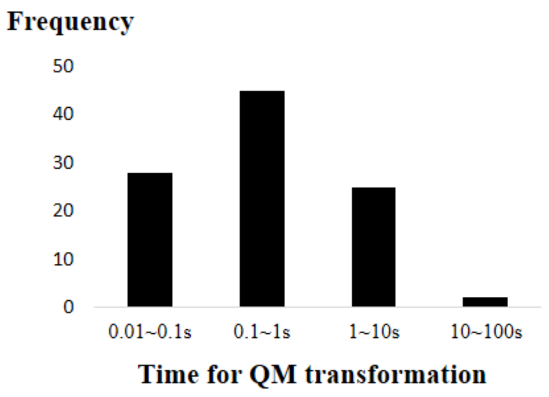


FIG.8. Histogram of QM transformation runtime on 100 randomly generated heterogeneous networks with

50 nodes. The result shows that the complexity of QM transformation is much less than identifying motifs.

Appendix B. Description of the multi-level Quine-McCluskey algorithm

Here we describe the implementation of the multi-level Quine-McCluskey algorithm:

```
1. Scan all functions to get the all states for each node.
2. For each function, enumerate all input combinations to get the minterms, make it list1
3. Group the implicants in list1 according to the number of zeroes
4. Compare between neighbor groups:
     For each implicant1 in group i:
       For each implicant2 in group i+1:
         If implicant1 and 2 are different by 1 digit:
           Access all implicants with all states of the different node, if they are all in group i+1, merge the implicants;
5. If an implicant does not get merged in any comparison, mark it. Go to step 4 with i+=1.
6. If there is no merged implicant, proceed to step 7. Otherwise set list1 to be the merged implicants, then go to step 3.
7. The marked implicants are prime implicants
8. Go to step 2 with the next function; repeat until all functions are transformed.
```

Appendix C. Mathematical foundations of the motif-based attractor identification algorithm

In this section we rigorously define the concepts we used in our motif-based method, and present important conclusions on why stable motifs and oscillating motifs can be used to find attractors. Our method does not depend on the update scheme, so the complex attractors predicted by our method are consistent with the complex attractors under an asynchronous update where one node is updated per time step. An efficient way to implement the most general case of asynchronous update is to randomly choose a node to update at each time step, which is the 'general asynchronous update' we mentioned in the main text. It is a representative update scheme for the broad class of update schemes where our method can accurately find all attractors.

Mathematical definitions of node states and regulatory functions

Let $v_i,\ i=(1,2,\dots,N)$ be the N nodes of a multi-level dynamical system; $m_i,\ i=(1,2,\dots N)$ be the highest level of node v_i (which means that it has *m*m_i*+1* levels, namely 0, 1… m_i). Let $\sigma_i,\ i=(1,2,\dots,N)$ be a state of the i^{th} node v_i; and $\Sigma=(\sigma_1,\sigma_2,\dots,\sigma_N)$ be a state of the entire system. We use Σ_P to represent a partial system state where $P=(\sigma_{m_1}=l_1,\sigma_{m_2}=l_2,\dots,\sigma_{m_M}=l_M)$, M < N is a subset of nodes that have their states specified, while the other states are unspecified.

Alternatively, we can represent the system with virtual nodes. We use $v_i^{(l)},\ l=(0,1,\dots,m_i)$ to represent the virtual node for the l^{th} state of v_i. The total number of virtual nodes is $N_v=\sum_{i=1}^{N}(m_i+1)$. $v_i^{(l)}$ is Boolean-like, meaning that it can only have state values 0 or 1. The state of each virtual node is now represented by $\sigma_i^{(l)},\ i=(1,2,\dots,N),\ l=(0,1,\dots,m_i)$. The state of the system is then represented as $\Sigma=\left(\sigma_1^{(1)},\sigma_1^{(2)},\dots,\sigma_1^{(m_1)},\sigma_2^{(1)},\sigma_2^{(2)},\dots,\sigma_N^{(m_N)}\right)$. Let $f_i\colon\aleph^N\to\{0,1,..m_i\}$ be the regulatory function of node v_i, where $\aleph^N$ is the potential state space of the system (as node levels are described by natural numbers); the actual state space has levels $0,1,..m_j$ for each node j. The regulatory function of each virtual node is a function of virtual nodes, e.g. the function of the i^{th} node's l^{th} state is $f_i^{(l)}\left(\sigma_{k_1}^{(l_1)},\sigma_{k_2}^{(l_2)},\dots\right)$ (where k_j is the j^{th} input of node i), thus it is Boolean-like, $f_i^{(l)}\colon\{0,1\}^{N_v}\to\{0,1\}$. Let $F=\left(f_1^{(0)},f_1^{(1)},\dots f_1^{(m_1)},f_2^{(0)},f_2^{(1)},\dots f_2^{(m_2)},\dots,f_N^{(m_N)}\right)$ be the vector of all virtual node functions. We use $f_i^{(l)}(\Sigma)$ to represent a function of a virtual node evaluated under state Σ of the system, and $f_i^{(l)}|_P$ to represent a function evaluated under a partial state P, where only $P=\left(\sigma_{p1}^{(0)},\sigma_{p1}^{(1)},\dots,\sigma_{p2}^{(0)},\sigma_{p2}^{(1)},\dots,\sigma_{pk}^{(0)},\sigma_{pk}^{(1)},\dots\right)$ are evaluated.

The virtual nodes that correspond to the same original node v_i are called 'sibling nodes' of each other, and these nodes form a sibling set of v_i, represented with $S_i=\left\{v_i^{(l)}\right\},l=(0,1,\dots,m_i)$. A sibling set satisfies the following property: when the functions of these nodes are evaluated based on a state Σ, one and only one of the functions in the set is 1 and the rest are 0, i.e. $\sum_{i=1}^{m_j}f_j^{(i)}(\Sigma)=1$, and $f_j^{(k)}(\Sigma)f_j^{(l)}(\Sigma)=0,\forall k\neq l$. When implemented in a simulation, all sibling virtual nodes corresponding to the same original node should be evaluated simultaneously.

We assume that each of the virtual nodes' regulatory functions has the following properties:
1. Non-constant. $f_i^{(l)}$ is not a constant, i.e. $f_i^{(l)} \neq 0$ and $f_i^{(l)} \neq 1$
2. Each input node is effective. If f_i depends on node v_j, then there must be at least one pair of network states $\Sigma^{(1)}$ and $\Sigma^{(2)}$ with $\sigma_j^{(1)} \neq \sigma_j^{(2)}$ and $\sigma_k^{(1)} = \sigma_k^{(2)}$ for all $k \neq j$ such that $f_i(\Sigma^{(1)}) \neq f_i(\Sigma^{(2)})$. Or, equivalently in terms of virtual nodes, if a sibling node function set $F_i = \{f_i^{(l)}\}$, $l = (0, 1, \ldots, m_i)$ depends on a set of sibling nodes S_i, then there must be at least one pair of network states $\Sigma^{(1)}$ and $\Sigma^{(2)}$ with $\sigma_j^{(1)} \neq \sigma_j^{(2)}$ and $\sigma_k^{(1)} = \sigma_k^{(2)}$ for all $k \neq j$, such that $\exists\, f_i^l(\Sigma^{(1)}) \neq f_i^l(\Sigma^{(2)})$.
3. Each Boolean-like function $f_i^{(l)}$ is in a disjunctive normal form (specifically, in a Blake canonical form), with the inputs being the virtual nodes:

$$f_i^{(l)} = \left(v_{j_1}^{(l_1)}\ and\ v_{j_2}^{(l_2)}\ and\ \ldots\ and\ v_{j_{c1}}^{(l_{c1})}\right)$$
$$or\ \left(v_{j_{c1+1}}^{(l_{c1+1})}\ and\ v_{j_{c1+2}}^{(l_{c1+2})}\ and\ \ldots\ and\ v_{j_{c2}}^{(l_{c2})}\right)\ or\ \ldots$$

In addition, if for a network state subset $P \subset \Sigma$, $f_i^{(1)}|_P = 1$ regardless of the states of the other nodes, then the disjunctive normal form of $f_i^{(l)}$ must have at least one conjunctive clause equal to 1 when evaluated under this partial state P.

Definition of the expanded network

The expanded network is a graph embodiment of the virtual nodes and their regulatory functions. The nodes of the expanded network consist of virtual nodes $v_i^{(l_j)}$, $i = (1,2,\ldots,N)$, $j = (1,2,\ldots,m_i)$ and composite nodes (which represent 'and' rules) $v_k^{(comp)}$, $(i = 1,2,\ldots,K)$, where K is the total number of 'and' rules used in the functions. The edges of the expanded network can be one of two types: edges from virtual or composite nodes to virtual nodes (which are aggregated with 'or' rules); and edges from virtual nodes to composite nodes (which are aggregated with 'and' rules). One can think of virtual nodes as having a function that contains only the Boolean operator 'or': $f_i^{(l)} = I_1\ or\ I_2\ or\ \ldots$, where the I's are inputs of the virtual node in the expanded network, including both virtual nodes and composite nodes. The composite nodes can be treated as having only the Boolean operator 'and': $f_i^{(comp)} = I_1\ and\ I_2\ and\ \ldots.$, where the I's are the inputs (virtual nodes) of the composite node. An example is provided in Sec. II E.

We define a sufficient regulator of a virtual node A as either a virtual node connected directly to A, or a composite node together with all of its input virtual nodes. Thus a sufficient regulator may be a group of virtual nodes.

Definitions of motifs

A **stable motif** is defined as a strongly-connected-component (SCC) of the expanded network that satisfies:
(1) If $v_i^{(l)}$ is in the SCC, then any $v_i^{(k)}$, $(k \neq l)$ is not in the SCC.
(2) If $v_k^{(comp)}$ is in the SCC, then all of its inputs are in the SCC.

An **oscillating motif** is defined as a strongly-connected-component (SCC) of the expanded network that satisfies:
(1) There exists a $v_i^{(l)}$ in the SCC such that at least one of its sibling nodes, say $v_i^{(k)}$, $(k \neq l)$ is also in the SCC.
(2) If $v_k^{(comp)}$ is in the SCC, then all of its inputs are in the SCC.

These motifs are described and illustrated in Sec. II F and G.

We also define a self-sufficient motif as an SCC in the expanded network that satisfies: If $v_k^{(comp)}$ is in the SCC, then all of its inputs are in the SCC. The intuition of this SCC is that it is a self-sustaining feedback loop. Stable motifs and oscillating motifs are special self-sufficient motifs, with extra requirements in the states involved in the motif.

It is important to note that both stable motifs and oscillating motifs correspond to SCCs in the original network. Stable motifs are SCCs in which all cycles are positive. Oscillating motifs contain negative cycles. These negative cycles may only be apparent when considering the specific regulatory functions.

In analogy to source nodes (i.e. nodes that do not have incoming edges), we call an SCC a source SCC if there are no nodes other than the nodes of the SCC that can reach the source SCC through directed paths.

There is a one-to-one correspondence between stable motifs and partial fixed points

We define a partial fixed point (or partial steady state), as a set of nodes and associated states in which the nodes stabilize regardless of the rest of the network. Note that this definition expresses a stricter condition than a set of nodes whose states stabilizes in a certain context (which depends on the rest of the network).

We show that each stable motif corresponds to a partial fixed point of the system, and that each partial fixed point corresponds to a stable motif.

Proposition 1. A stable motif corresponds to a fixed point of the nodes that participate in the motif, i.e. the states of the nodes of the stable motif remain the same regardless of the state of the other nodes. Formally,

Let $M = \left(v_{j_1}^{(l_1)}, v_{j_2}^{(l_2)}, \dots, v_{j_k}^{(l_k)}, v_{m_1}^{(comp)}, v_{m_2}^{(comp)}, \dots, v_{m_L}^{(comp)}\right)$ *be a stable motif where* $v_{j_1}^{(l_1)}, v_{j_2}^{(l_2)}, \dots, v_{j_k}^{(l_k)}$ *are virtual nodes and* $v_{m_1}^{(comp)}, v_{m_2}^{(comp)}, \dots, v_{m_L}^{(comp)}$ *are composite nodes. Let* $P = (\sigma_{j_1} = l_1, \sigma_{j_k} = l_k, \dots, \sigma_{j_k} = l_k)$ *be a partial system state. Then for any system state* Σ_P *with* $\sigma_{j_i} = l_i$ *, we have* $f_{j_i}^{(l_k)}(\Sigma_P) = \delta_{ik}$.

Sketch of proof: We first show that $f_{j_i}^{(l_i)}(\Sigma_P) = 1$. By definition of a stable motif, each virtual node's function must have a conjunctive clause (implicant) that consists of either of the following: (1) a virtual node of the same stable motif; or (2) a composite node whose inputs consists only of virtual nodes of the same stable motif. This implicant will be 1 when $f_{j_i}^{(l_i)}(\Sigma_P)$ is evaluated, fixing the value $f_{j_i}^{(l_i)}(\Sigma_P) = 1$. Then $f_{j_i}^{(l_k)}(\Sigma_M) = 0\ \forall k \neq i$ is trivially true because the functions of sibling nodes must satisfy: $f_j^{(k)}(\Sigma) f_j^{(l)}(\Sigma) = 0\ \forall k \neq l$.

Proposition 2. (Reverse of proposition 1) For any partial fixed point of the system, i.e. a set of node states where updating any involved node gives back the same state for the node, there is a set of stable motifs that correspond to it. Formally,

Let $P = (\sigma_{j_1} = l_1, \sigma_{j_k} = l_k, \dots, \sigma_{j_k} = l_k)$ *be a partial system state such that* $f_{j_i}^{(l_i)}(\Sigma_P) = 1, \forall j_i$. *Then (1) there exists a set of stable motifs* $\{M_n\}$ *where each stable motif contains only nodes from* $\{v_{j_i}^{l_i}\}, i = 1, \dots, k$ *as virtual nodes; (2) the nodes specified in P but not in nodes of* $\{M_n\}$ *are downstream of the nodes of* $\{M_n\}$.

Sketch of proof: From the disjunctive normal form of the functions, $f_{j_i}^{(l_i)}(\Sigma_P) = 1$ means that at least one of the conjunctive clauses of each function is 1, and consists of virtual nodes specified in P. Then one can create a sub-network of the expanded network, whose nodes are these virtual nodes as well as composite nodes representing conjunctive clauses; and edges are added if a virtual node or composite node is an input in a virtual node's function, or if a virtual node is an input of a composite node. Since each virtual node in this sub-network has at least one input within the sub-network, there exists at least one SCC. This SCC(s) is/are the stable motif(s) we are looking for.

Stable and oscillating parts of complex attractors

A complex attractor of the whole system consists of a set of states that the system keeps revisiting. When considering the states visited by each node in a complex attractor, there may be a subset of nodes whose state remains the same. We call these nodes stabilized nodes. The remaining nodes (potentially, all nodes) will oscillate, meaning that they will keep revisiting all, or possibly a subset, of their states. We will call these nodes oscillating nodes. In the following two propositions we establish the relationships between these nodes.

Proposition 3. Stabilized nodes in an attractor can be downstream of stabilized nodes or downstream of oscillating nodes.

Let A be an attractor of a multi-level dynamical system under general asynchronous update, and let S and O be the stabilized and oscillating nodes, respectively. If $v_s \subset S$ *and* l_s *is the node's stabilized value, then one of the following holds: (1) one of the conjunctive clauses of* $f_s^{(l_s)}$ *depends only on nodes of S in A; if (1) is not true, then (2)* $f_s^{(l_s)}$ *and the function of at least one sibling node* $f_s^{(k_s)}, k_s \neq l_s$ *have at least one conjunctive clause dependent on the nodes in O.*

The first case is self-evident. An example for the second case is a network with Boolean nodes, A, B and C:

$f_A^{(0)} = (A1 \text{ or } B1)\ and\ C0,$
$f_A^{(1)} = A0\ and\ B0\ or\ C1,$
$f_B^{(0)} = (A1 \text{ or } B1)\ and\ C0,$
$f_B^{(1)} = A0\ amd\ B0\ or\ C1,$
$f_C^{(0)} = B0\ and\ C0\ or\ A0\ and\ C0,$
$f_C^{(1)} = (A1\ and\ B1)\ or\ C1,$

where for simplicity the virtual nodes are denoted Xi, X={A,B,C} instead of Xi. This network has an oscillating attractor with A and B oscillating and

C stabilized at 0. C is stabilized despite being regulated by nodes that oscillate. It does not satisfy (1) in the proposition; instead, $f_C^{(0)}$ and $f_C^{(1)}$ satisfy (2) in the proposition.

Proposition 4. Oscillating nodes in an attractor must be downstream of oscillating nodes.

Let A be an attractor of a multi-level dynamical system under general asynchronous update, and let S and O be the stabilized and oscillating nodes, respectively. If $v_O \subset O$ and $l_{O_1}, l_{O_2}, \dots, l_{O_k}$ are the oscillating states, then the following holds: none of the conjunctive clauses of $f_{O_i}^{(l_{O_i})}, (i = 1,2,\dots,k)$ depends only on nodes of S in A; or alternatively, all functions $f_{O_i}^{(l_{O_i})}, (i = 1,2,\dots,k)$ have at least one conjunctive clause dependent on state of nodes in O.

The proof for proposition 4 is straightforward.

Iterative stable motif based network reduction conserves the attractors of the system

We proceed to the proof of conservation of attractors during iterative network reduction by stating three lemmas.

Lemma 1. Construction of the stabilized set S_{red} that corresponds to at least one stable motif

Let A be an attractor of a multi-level dynamical system under general asynchronous update, and let S and O be the stabilized and oscillating nodes, respectively. If there is a partial fixed point in A, then: there exists a set of nodes $S_{red} \subset S$ such that in the expanded network representation there will be at least one stable motif composed only of virtual nodes of S_{red} in A, or composite nodes composed of such nodes.

Sketch of proof: Each stabilized node in S corresponds to a function $f_S^{(l_s)}$. By Proposition 3, we can divide S into nodes whose functions have a conjunctive clause that depends only on node states (virtual nodes) specified in S, denoted S_0, and nodes that have at least one conjunctive clause in their rule dependent on the states of nodes in O, denoted S_{osc}. Let $S_1 \subset S_0$ be the nodes that have at least one conjunctive clause dependent only on nodes' states specified in S_0. Let $S_2 \subset S_1$ be the nodes that have at least one conjunctive clause dependent only on node states specified in S_1. One can do this iteratively until $S_{i_{max}} = S_{i_{max}+1}$, and denote $S_{red} = S_{i_{max}}$. Since there exists a partial fixed point, S_{red} will contain nodes in the partial fixed point and will not be an empty set. The iterative selection guarantees that S_{red} does not depend on oscillating nodes or nodes influenced by oscillating nodes. And since the function of each node in S_{red} contains at least one conjunctive clause dependent only on nodes in S_{red} itself, there is at least one SCC in S_{red} and this SCC satisfies the definition of a stable motif.

Lemma 2. Network reduction based on stable motifs stabilizes the nodes in S_{red}

Let $S_{red} \subset S$ be the set of nodes constructed in Lemma 1. Then (1) Network reduction based on stable motifs composed only of nodes from S_{red} can only stabilize nodes in S_{red}. Moreover, (2) if a node i in S_{red} stabilizes during the reduction, it has to stabilize at its state specified in A; if a node i does not stabilize during the reduction, then after the reduction, its function $f_i^{(l_s)}$, where l_s is the node's stabilized state in A, must have a conjunctive clause that depends only on nodes' states specified in S_{red} in A that did not stabilize during reduction.

Sketch of proof: We first prove (1) by showing that the other nodes, i.e. nodes in $S_0 - S_{red}$ and S_{osc}, cannot stabilize from stable motifs composed only of nodes from S_{red}. This statement is straightforward from the definitions of $S_0 - S_{red}$ and S_{osc}. Nodes in $S_0 - S_{red}$ do not have any conjunctive clauses that depend only on nodes' states from S_{red}, otherwise the nodes would be in S_{red}. According to Proposition 3, nodes in S_{osc} do not have any conjunctive clauses that depend only on nodes' states from S_{red}. Therefore reduction based on stable motifs composed only of nodes from S_{red} is not sufficient to stabilize these nodes. To show (2), consider the iterative process of reduction by plugging in the stabilized nodes' states. One starts with a chosen SCC in S_{red}, and then nodes with at least one conjunctive clause depending only on nodes states from S_{red} will stabilize in their value in A. When this reduction is applied iteratively until it cannot be done anymore, the resulting S_{red} contains only non-stabilized nodes, whose functions do not have any dependence on the reduced nodes. Then these functions must have a conjunctive clause that depends only on nodes' states specified in S_{red} in A that did not stabilize during reduction.

Lemma 3. In a system/reduced system with no stable motifs, all nodes are influenced by oscillating nodes.

Let A be an attractor of a multi-level dynamical system under general asynchronous update, and *let S and O be the stabilized and oscillating nodes, respectively. Let $S_{red} \subset S$ be the set of nodes constructed in Lemma 1. Assume S_{red} is empty and O is not empty. Then in the original system, all nodes in O and S must all be a part of, or downstream of, a set of source SCCs, each of which contains at least one oscillating motif. Moreover, the oscillating motifs will contain the virtual nodes corresponding to all the states visited by the oscillating nodes.*

Sketch of proof: We can assume that there are no source nodes in the network corresponding to the dynamical system, because if there are any, one can reduce them and substitute their values of the source nodes into the regulatory functions of their downstream nodes. The network contains one or more source SCCs. Then, any source SCC in the network must contain at least one oscillating node, otherwise this source SCC would contain only stabilized nodes, meaning a non-empty S_{red}.

We then show that any of these source SCCs corresponds to at least one oscillating motif in the expanded network. Suppose that a pair of sibling virtual nodes $v_1^{(l_1)}, v_1^{(l_2)}$ correspond to an oscillating node v_1 in the source SCC. Since it is a source SCC, all regulators of v_1 are from this SCC, and v_1 regulates at least one other node from this SCC. Consider the expanded network around $v_1^{(l_1)}$. We construct an oscillating motif candidate starting with marking its regulators and selected targets. First we mark all inputs of $v_1^{(l_1)}$, including inputs directly connected to $v_1^{(l_1)}$ and inputs connected to $v_1^{(l_1)}$ via composite nodes. All marked virtual nodes correspond to nodes in the source SCC. Then we mark the target virtual nodes of $v_1^{(l_1)}$ that satisfy: (1) the target is regulated directly by $v_1^{(l_1)}$ or via one composite node; (2) the target corresponds to a node in the source SCC. We iteratively continue this marking process for all marked virtual nodes. Since in each step only virtual nodes corresponding to nodes in the source SCC are marked, and each node marked must have at least one regulator and one selected target, we will obtain an SCC in the expanded network all of whose virtual nodes correspond to the source SCC in the original graph. Because we started the process in a source SCC in the original network, if a composite node is marked, all of its inputs will satisfy the marking condition, and will be marked as well. We refer to this SCC in the expanded network as the expanded motif, and will show that it can be used to construct an oscillating motif. Notice that for both $v_1^{(l_1)}$ and $v_1^{(l_2)}$, one can construct the corresponding expanded motif, respectively. Because this pair of virtual nodes represents oscillating states under a general asynchronous complex attractor, they must be connected to each other, otherwise they cannot oscillate. Thus their expanded motifs are strongly connected, and can be merged to obtain a larger strongly connected motif that includes both $v_1^{(l_1)}$ and $v_1^{(l_2)}$. In cases where more than two virtual nodes corresponding to the same node are involved in an oscillation, the same merging can be applied, and it similarly results in a single expanded motif. This merging can be done for each pair of oscillating sibling nodes. The resulting merged motif is an oscillating motif, because the marking process guarantees that all inputs of composite nodes are marked; and the merging guarantees that at least two states of oscillating nodes are marked. In addition, all oscillating virtual nodes in the oscillation are marked, i.e. the oscillating motif covers all the oscillating states of each oscillating node in the oscillation.

Therefore, after the reduction of stable motifs, in a reduced network any source SCC corresponds to at least one oscillating motif, and all nodes in the expanded network are either part of an oscillating motif or downstream of an oscillating motif.

Remark: It is worth pointing out that complex attractors of a dynamic model depend on the update scheme. Some complex attractors only exist if a specific update scheme is imposed (see Appendix D1). Therefore, a timing-independent method like ours is not able to find candidates of all complex attractors, but only candidates for timing-independent complex attractors, i.e. complex attractors under asynchronous update. In the proof of Lemma 3, this is reflected by the condition "Because this pair of virtual nodes represents oscillating states under a general asynchronous complex attractor, they must be connected to each other, otherwise they cannot oscillate." Everything else in the proof applies for arbitrary update schemes. In addition, the actual oscillation may be different from the corresponding oscillating motifs, so no exact conclusions can be made regarding nodes downstream of an oscillating motif.

The following theorem is the main result of this section, and it combines the results of Lemma 1, 2, and 3. It shows that for every attractor of the system, our motif-based method will find a corresponding quasi-attractor in which:

(1) The state of the nodes in S_{red} is the same as in the attractor
(2) There is at least one oscillating motif that corresponds to the oscillating part of each complex attractor.

Theorem 1. Conservation of attractors in motif reduction

Let A be an attractor of a multi-level dynamical system under general asynchronous update, and let S and O be the stabilized and oscillating nodes, respectively. Let $S_{red} \subset S$ be the set of nodes constructed in Lemma 1. Then, there exists a set of stable motifs such that, by applying network reduction, all the nodes in S_{red} will stabilize in their steady state in A, while the rest of the nodes will be part of the final reduced network. This final reduced network will be such that all nodes in O and S must all be a part of, or downstream of a set of source SCCs, each of which contains at least one oscillating motif. Moreover, the oscillating motifs will contain the virtual nodes corresponding to all the states visited by the oscillating nodes.

Sketch of proof: Using Lemma 2, the network obtained after reducing any stable motif composed only of the corresponding states of S_{red} in A will have a new S_{red} containing only the nodes in the previous S_{red} that did not stabilize. One can iteratively plug in the stable motifs until S_{red} is empty. Because of Lemma 1, there is always a stable motif as long as S_{red} is not empty. In the reduction process only nodes in S_{red} can stabilize. By Lemma 3, the source SCCs in the resulting reduced network contains oscillating motifs that cover all virtual nodes corresponding to oscillating states of oscillating nodes.

Finally we list some straightforward corollaries of the theorem that help demonstrate the properties of attractors.

Corollary 1. If a multi-level dynamic system does not have oscillating motifs in its expanded network, the system does not have complex attractors.

Corollary 2. If a multi-level system does not have fixed point attractors, it must have at least one oscillating motif.

Corollary 3. A quasi-attractor can correspond to multiple complex attractors. Examples in Appendix D illustrate this corollary.

Appendix D. Oscillating Motif Examples

Here we illustrate certain properties of oscillating motifs with examples. Because certain regulatory relationships between nodes are non-monotonic (their sign depends on the node state), for simplicity we use the same type of arrow for all edges. For better visualization, we omitted the names of composite nodes in complicated expanded networks.

1. Timing-dependent complex attractor

Figure 9 shows an example of a dynamical system with different attractors under different update schemes.

In synchronous update al nodes are updated simultaneously, thus state transitions are deterministic. Each state has only one successor (i.e. each node of the state transition graph has a single outgoing edge). In the state transition graph corresponding to general asynchronous update, a given state has as many potential state transitions as many nodes there are in the system (because each node has a chance to be updated).

In this example a complex attractor exists for synchronous update, but not for general asynchronous update. This complex attractor is induced by positive feedback, not negative feedback, and requires that nodes A and B are updated at exactly the same time. So it is timing-dependent and will not be preserved under fluctuations in timing. This type of timing-dependent complex attractor will not be identified by our motif-based method.

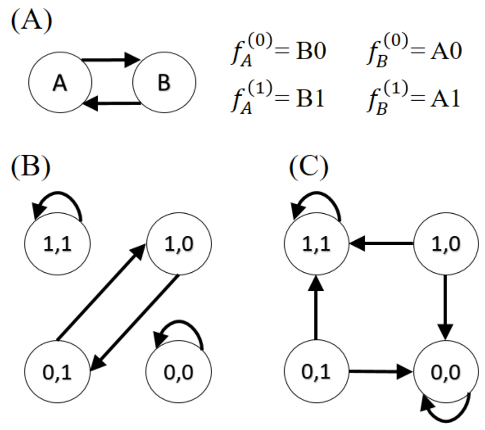


FIG. 9. An example of a timing-dependent complex attractor. (A) The network and regulatory functions. (B) The state transition graph under synchronous update. Each node of the state transition graph is a state, given in the order A, B, and each edge is a state transition allowed by synchronous update. The system has two fixed points, (0,0) and (1,1). It also has a complex attractor formed by the states (0,1) and (1,0). (C) The state transition graph under general asynchronous

update (i.e. when one node is updated at a time). Only the two fixed point attractors exist. The synchronous complex attractor is timing-dependent and does not exist in this update scheme.

2. The existence of an oscillating motif does not guarantee the existence of a complex attractor

Figure 10 demonstrates a simple example where the oscillating motif corresponds to a transient oscillation, which will converge into a fixed point attractor.

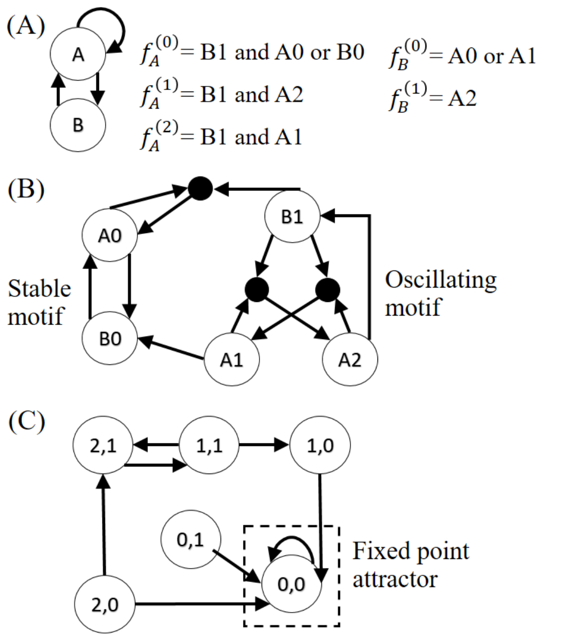


FIG. 10. An example of an oscillating motif without a complex attractor. (A) The network and regulatory functions. (B) The expanded network and motifs. There is a stable motif formed by A0 and B0, and an oscillating motif made up by A1, A2, B1. (C) The state transition graph using general asynchronous update. There is only one attractor, which is a fixed point. The transient oscillation between states (2,1) and (1,1) will eventually converge into the fixed point.

3. Oscillating nodes can have stabilized downstream nodes

Figure 11 shows a Boolean example adapted from [30] in (A)(B) and a multi-level example in (C). In the system on Figure 11(A), nodes A and B do not visit the state A=1, B=1 unless starting from there, which causes the stabilization of C=0. Such situations are expected to be more common in multi-level systems than in Boolean systems. In the system of Figure 11(C) the regulator node A has more states than the regulated node B, thus the oscillation in A does not affect B This situation is expected to be observed in biological systems.

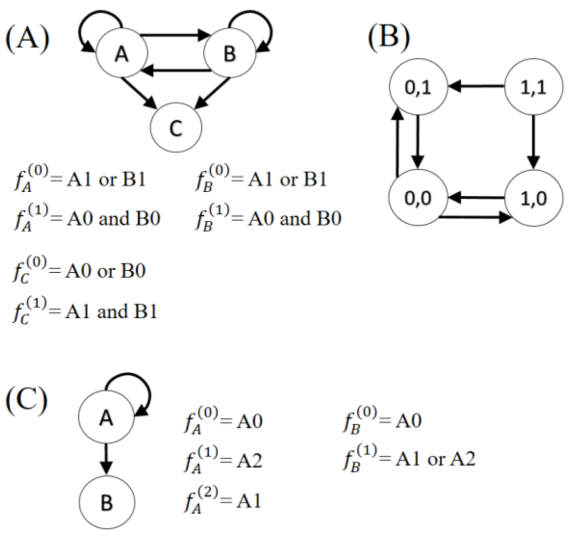


FIG. 11. Examples of stabilized nodes downstream of oscillating node(s). (A) A Boolean example where A and B oscillate but their downstream C is stable under that oscillation. (B) The general asynchronous state transition graph of nodes A and B. The state (A=1,B=1) is not visited in the long term, leading to the stabilization of C=0. (C) A multi-level example where A is oscillating between 1 and 2, leading to B stabilizing at 1. This example arises because of asymmetry in the nodes' number of states: A has three states but B only has two states.

4. Co-existence of a fixed point and a complex attractor

If a dynamical system has input variables (source nodes with sustained states), it can have a different attractor for different values of the input variables. Here we consider a dynamical system with a given choice of input variables, or equivalently, no input variables. Co-existence of a fixed point attractor and a complex attractor for such a system is possible but rare in Boolean systems. Zañudo et al. [30] referred to this situation as unstable oscillation. We reproduce the example given in as Figure 12. Notice that the nodes involved in the two attractors share node states, i.e. A is fixed at 1 in the fixed point attractor, but also enters state 1 in the complex attractor. In multi-level dynamical systems the fixed point and complex attractor do not need to share node states (see Figure 5 and Figure 6 in section II-G). Thus we expect that coexistence of (potentially multiple) fixed point(s) and complex attractor(s) is more frequently observed.

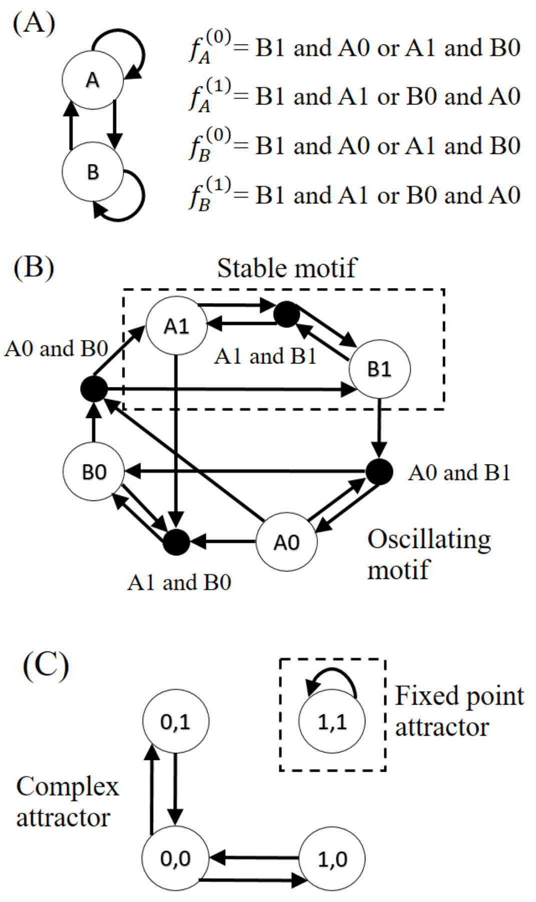

(A)
A
B
$f_A^{(0)}$= B1 and A0 or A1 and B0
$f_A^{(1)}$= B1 and A1 or B0 and A0
$f_B^{(0)}$= B1 and A0 or A1 and B0
$f_B^{(1)}$= B1 and A1 or B0 and A0
(B)
Stable motif
A0 and B0
A1
A1 and B1
B1
B0
A0 and B1
A0
Oscillating motif
A1 and B0
(C)
0,1
1,1
Fixed point attractor
Complex attractor
0,0
1,0

FIG. 12. An example of an unstable oscillation. The system has a fixed point and a complex attractor. (A) The network and regulatory functions. (B) The expanded network and motifs. The entire expanded network forms an oscillating motif, containing the stable motif by two nodes A1, B1, and one composite node. (C) The state transition graph using general synchronous update. There is a fixed point attractor A=1, B=1, and a complex attractor. Note that in the complex attractor, although both A and B are allowed to enter state 1, they cannot be in state 1 simultaneously.

5. One oscillating motif can correspond to multiple attractors

Figure 12 also illustrates that the same oscillating motif can correspond to multiple attractors, in this case a complex attractor and a fixed point. In multi-level cases, multiple complex attractors can also be found within the same oscillating motif. Figure 13 shows such an example. Combined with the property that an oscillating motif does not guarantee a complex attractor, the conclusion is that there is no exact match between the actual number of complex attractors and the number of quasi-attractors found, i.e. there may be more actual attractors than quasi-attractors found, and there may be less actual attractors than quasi-attractors found.

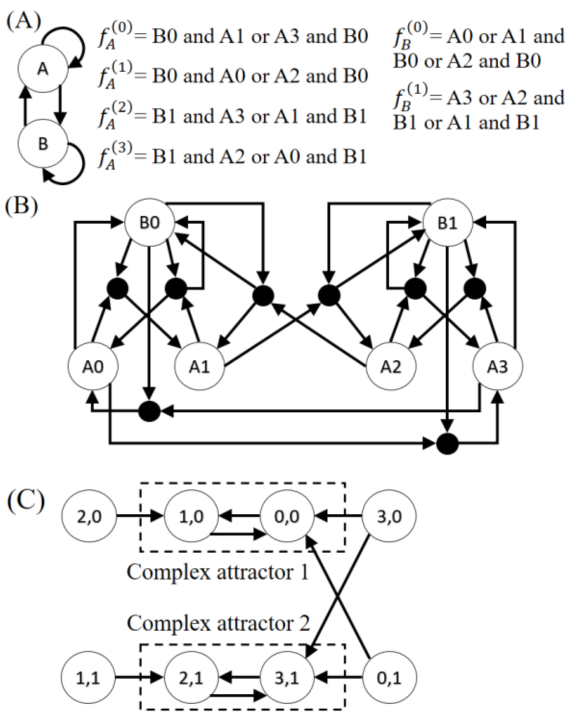

(A)
A
B
$f_A^{(0)}$= B0 and A1 or A3 and B0
$f_A^{(1)}$= B0 and A0 or A2 and B0
$f_A^{(2)}$= B1 and A3 or A1 and B1
$f_A^{(3)}$= B1 and A2 or A0 and B1
$f_B^{(0)}$= A0 or A1 and B0 or A2 and B0
$f_B^{(1)}$= A3 or A2 and B1 or A1 and B1
(B)
B0
B1
A0
A1
A2
A3
(C)
2,0
1,0
0,0
3,0
Complex attractor 1
Complex attractor 2
1,1
2,1
3,1
0,1

FIG. 13. An example of an oscillating motif containing two complex attractors. (A) The network and regulatory functions. (B) The expanded network and motifs. The entire expanded network forms an oscillating motif. (C) The state transition graph. For simplicity self-loops representing self-transitions are not shown in the graph. There are two complex attractors, the first attractor is B=0, A=0 or 1, and the second attractor is B=1, A =2 or 3.

Appendix E. Generation of regulatory functions in synthetic networks

Here we describe how we randomly generated regulatory functions among those consistent with the number of regulators and number of states for each node.

In the network generation part, each node's regulators are generated. In the benchmarks, we generated networks where each node has two input nodes. For each target node, we assign to each combination of different states of the regulator nodes a randomly selected state of the target node. For example, if Boolean target node A is regulated by Boolean nodes B and C, each of the four state combinations of B and C will be randomly assigned to either the function of A0 or A1. Different input combinations assigned to the same target state will be separated by an '*or*' operator. For example, combinations B0 C0 and B1 C0 are assigned to A0, then the function of A0 is just $f_A^{(0)}$ = (B0 *and* C0) *or* (B1 *and* C0). If at the end of the assignment a target state did not get any assigned combination, this function is ineffective, and we discard all the functions of this target node and start over to generate a new set of functions.

References

[1] A. Arenas, A. Díaz-Guilera, J. Kurths, Y. Moreno, C. Zhou, Synchronization in complex networks, Physics Reports, 469 (2008) 93-153.
[2] X.-J. Tian, H. Zhang, J. Sannerud, J. Xing, Achieving diverse and monoallelic olfactory receptor selection through dual-objective optimization design, Proceedings of the National Academy of Sciences, 113 (2016) E2889-E2898.
[3] A.-L. Barabasi, Z.N. Oltvai, Network biology: understanding the cell's functional organization, Nat Rev Genet, 5 (2004) 101-113.
[4] D. Deritei, W.C. Aird, M. Ercsey-Ravasz, E.R. Regan, Principles of dynamical modularity in biological regulatory networks, Sci Rep, 6 (2016) 21957.
[5] J.J. Tyson, K. Chen, B. Novak, Network dynamics and cell physiology, Nature Reviews Molecular Cell Biology, 2 (2001) 908-916.
[6] R. Albert, R.S. Wang, DISCRETE DYNAMIC MODELING OF CELLULAR SIGNALING NETWORKS, in: M.L. Johnson, L. Brand (Eds.) Methods in Enzymology: Computer Methods, Part B, Elsevier Academic Press Inc, San Diego, 2009, pp. 281-306.
[7] M. Pennisi, S. Cavalieri, S. Motta, F. Pappalardo, A methodological approach for using high-level Petri Nets to model the immune system response, BMC bioinformatics, 17 (2016) 498.
[8] A.A. Butchy, N. Miskov-Zivanov, Discrete modeling of macrophage differentiation, The Journal of Immunology, 198 (2017) 67.13-67.13.
[9] R. Albert, J. Thakar, Boolean modeling: a logic-based dynamic approach for understanding signaling and regulatory networks and for making useful predictions, Wiley interdisciplinary reviews. Systems biology and medicine, 6 (2014) 353-369.
[10] R. Zhang, M.V. Shah, J. Yang, S.B. Nyland, X. Liu, J.K. Yun, R. Albert, T.P. Loughran, Jr., Network model of survival signaling in large granular lymphocyte leukemia, Proc Natl Acad Sci U S A, 105 (2008) 16308-16313.
[11] F. Li, T. Long, Y. Lu, Q. Ouyang, C. Tang, The yeast cell-cycle network is robustly designed, Proceedings of the National Academy of Sciences of the United States of America, 101 (2004) 4781-4786.
[12] W. Abou-Jaoude, P. Traynard, P.T. Monteiro, J. Saez-Rodriguez, T. Helikar, D. Thieffry, C. Chaouiya, Logical Modeling and Dynamical Analysis of Cellular Networks, Frontiers in genetics, 7 (2016) 94.
[13] S. Havlin, D.Y. Kenett, E. Ben-Jacob, A. Bunde, R. Cohen, H. Hermann, J.W. Kantelhardt, J. Kertész, S. Kirkpatrick, J. Kurths, J. Portugali, S. Solomon, Challenges in network science: Applications to infrastructures, climate, social systems and economics, The European Physical Journal Special Topics, 214 (2012) 273-293.
[14] J.-P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, A.-L. Barabási, Structure and tie strengths in mobile communication networks, Proceedings of the National Academy of Sciences, 104 (2007) 7332-7336.
[15] B. Federico, P. Matjaž, L. Vito, Determinants of public cooperation in multiplex networks, New Journal of Physics, 19 (2017) 073017.
[16] A. Lancichinetti, S. Fortunato, J. Kertész, Detecting the overlapping and hierarchical community structure in complex networks, New Journal of Physics, 11 (2009) 033015.
[17] F. Mori, A. Mochizuki, Expected Number of Fixed Points in Boolean Networks with Arbitrary Topology, Physical Review Letters, 119 (2017) 028301.
[18] R. Thomas, European Molecular Biology Organization., Kinetic logic : a Boolean approach to the analysis of complex regulatory systems : proceedings of the EMBO course "Formal analysis of genetic regulation," held in Brussels, September 6-16, 1977, Springer-Verlag, Berlin ; New York, 1979.
[19] H. Klarner, A. Bockmayr, H. Siebert, Computing maximal and minimal trap spaces of Boolean networks, Natural Computing, 14 (2015) 535-544.
[20] A. Garg, A. Di Cara, I. Xenarios, L. Mendoza, G. De Micheli, Synchronous versus asynchronous modeling of gene regulatory networks, Bioinformatics, 24 (2008) 1917-1925.
[21] A. Naldi, D. Thieffry, C. Chaouiya, Decision Diagrams for the Representation and Analysis of Logical Models of Genetic Networks, in: M. Calder, S. Gilmore (Eds.) Computational Methods in Systems Biology: International Conference CMSB 2007, Edinburgh, Scotland, September 20-21, 2007. Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 233-247.
[22] R. Laubenbacher, F. Hinkelmann, D. Murrugarra, A. Veliz-Cuba, Algebraic Models and Their Use in Systems Biology, in: N. Jonoska, M. Saito (Eds.) Discrete and Topological Models in Molecular Biology, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 443-474.
[23] P. Traynard, A. Fauré, F. Fages, D. Thieffry, Logical model specification aided by model-checking techniques: application to the mammalian cell cycle regulation, Bioinformatics, 32 (2016) i772-i780.
[24] J. Gómez Tejeda Zañudo, M. Scaltriti, R. Albert, A network modeling approach to elucidate drug resistance mechanisms and predict combinatorial drug treatments in breast cancer, Cancer Convergence, 1 (2017) 5.
[25] Z. Sun, X. Jin, R. Albert, S.M. Assmann, Multi-level modeling of light-induced stomatal opening offers new insights into its regulation by drought, PLoS Comput Biol, 10 (2014) e1003930.

[26] J. Chifman, S. Arat, Z. Deng, E. Lemler, J.C. Pino, L.A. Harris, M.A. Kochen, C.F. Lopez, S.A. Akman, F.M. Torti, S.V. Torti, R. Laubenbacher, Activated Oncogenic Pathway Modifies Iron Network in Breast Epithelial Cells: A Dynamic Modeling Perspective, PLoS computational biology, 13 (2017) e1005352.
[27] E. Dubrova, M. Liu, M. Teslenko, Finding Attractors in Synchronous Multiple-Valued Networks Using SAT-based Bounded Model Checking, Journal of Multiple-Valued Logic and Soft Computing, 19 (2012) 109-131.
[28] F. Hinkelmann, M. Brandon, B. Guang, R. McNeill, G. Blekherman, A. Veliz-Cuba, R. Laubenbacher, ADAM: Analysis of Discrete Models of Biological Systems Using Computer Algebra, BMC bioinformatics, 12 (2011) 295.
[29] C. Chaouiya, A. Naldi, D. Thieffry, Logical modelling of gene regulatory networks with GINsim, Methods Mol Biol, 804 (2012) 463-479.
[30] J.G. Zanudo, R. Albert, An effective network reduction approach to find the dynamical repertoire of discrete dynamic networks, Chaos, 23 (2013).
[31] B.L. Puniya, L. Allen, C. Hochfelder, M. Majumder, T. Helikar, Systems Perturbation Analysis of a Large-Scale Signal Transduction Model Reveals Potentially Influential Candidates for Cancer Therapeutics, Frontiers in Bioengineering and Biotechnology, 4 (2016) 10.
[32] L. Glass, S.A. Kauffman, LOGICAL ANALYSIS OF CONTINUOUS, NONLINEAR BIOCHEMICAL CONTROL NETWORKS, Journal of Theoretical Biology, 39 (1973) 103-129.
[33] X. Cheng, M. Sun, J.E.S. Socolar, Autonomous Boolean modelling of developmental gene regulatory networks, Journal of the Royal Society Interface, 10 (2013) 20120574.
[34] D. Murrugarra, A. Veliz-Cuba, B. Aguilar, S. Arat, R. Laubenbacher, Modeling stochasticity and variability in gene regulatory networks, EURASIP Journal on Bioinformatics and Systems Biology, 2012 (2012) 5.
[35] M. Chaves, R. Albert, E.D. Sontag, Robustness and fragility of Boolean models for genetic regulatory networks, J Theor Biol, 235 (2005) 431-449.
[36] R. Thomas, Regulatory networks seen as asynchronous automata: A logical description, Journal of theoretical biology, 153 (1991) 1-23.
[37] A. Saadatpour, I. Albert, R. Albert, Attractor analysis of asynchronous Boolean models of signal transduction networks, J Theor Biol, 266 (2010) 641-656.
[38] E. Remy, P. Ruet, D. Thieffry, Graphic requirements for multistability and attractive cycles in a Boolean dynamical framework, Advances in Applied Mathematics, 41 (2008) 335-350.
[39] A. Richard, Negative circuits and sustained oscillations in asynchronous automata networks, Advances in Applied Mathematics, 44 (2010) 378-392.
[40] K. Klemm, S. Bornholdt, Topology of biological networks and reliability of information processing, Proceedings of the National Academy of Sciences of the United States of America, 102 (2005) 18414-18419.
[41] X. Gan, R. Albert, A general method to find the attractors of discrete dynamic models of biological systems., in: the 8th International Conference on Physics and Control (PhysCon 2017), Florence, Italy, 2017.
[42] F.M. Brown, The Blake Canonical Form, in: Boolean Reasoning: The Logic of Boolean Equations, Springer US, Boston, MA, 1990, pp. 71-86.
[43] W.V. Quine, The Problem of Simplifying Truth Functions, The American Mathematical Monthly, 59 (1952) 521-531.
[44] W.V. Quine, A Way to Simplify Truth Functions, The American Mathematical Monthly, 62 (1955) 627-631.
[45] E.J. McCluskey, Minimization of Boolean Functions*, Bell System Technical Journal, 35 (1956) 1417-1444.
[46] A. Saadatpour, R. Albert, T.C. Reluga, A Reduction Method for Boolean Network Models Proven to Conserve Attractors, Siam Journal on Applied Dynamical Systems, 12 (2013) 1997-2011.
[47] A. Naldi, E. Remy, D. Thieffry, C. Chaouiya, Dynamically consistent reduction of logical regulatory graphs, Theoretical Computer Science, 412 (2011) 2207-2218.
[48] J.G. Zanudo, R. Albert, Cell fate reprogramming by control of intracellular network dynamics, PLoS Comput Biol, 11 (2015) e1004193.
[49] S.A. Kauffman, Metabolic stability and epigenesis in randomly constructed genetic nets, Journal of theoretical biology, 22 (1969).
[50] M. Aldana, S. Coppersmith, L.P. Kadanoff, Boolean Dynamics with Random Couplings, in: E. Kaplan, J.E. Marsden, K.R. Sreenivasan (Eds.) Perspectives and Problems in Nolinear Science: A Celebratory Volume in Honor of Lawrence Sirovich, Springer New York, New York, NY, 2003, pp. 23-89.
[51] R.S. Wang, R. Albert, Effects of community structure on the dynamics of random threshold networks, Physical Review E, 87 (2013).
[52] X. Gan, R. Albert, Analysis of a dynamic model of guard cell signaling reveals the stability of signal propagation, BMC systems biology, 10 (2016) 78.
[53] D. Berenguier, C. Chaouiya, P.T. Monteiro, A. Naldi, E. Remy, D. Thieffry, L. Tichit, Dynamical modeling and analysis of large cellular regulatory networks, Chaos, 23 (2013) 025114.
[54] See Supplemental Material S1 at [URL will be inserted by publisher] for details of the tests on biological models.
[55] W. Reisig, Petri Nets, in: I. Koch, W. Reisig, F. Schreiber (Eds.) Modeling in Systems Biology: The Petri Net Approach, Springer London, London, 2011, pp. 37-56.

[56] C. Chaouiya, A. Naldi, E. Remy, D. Thieffry, Petri net representation of multi-valued logical regulatory graphs, Natural Computing, 10 (2011) 727-750.
[57] R. Samaga, S. Klamt, Modeling approaches for qualitative and semi-quantitative analysis of cellular signaling networks, Cell Communication and Signaling, 11 (2013) 43.
[58] D.B. Johnson, Finding All the Elementary Circuits of a Directed Graph, SIAM Journal on Computing, 4 (1975) 77-84.
[59] R.-S. Wang, R. Albert, Elementary signaling modes predict the essentiality of signal transduction network components, BMC systems biology, 5 (2011) 44.
[60] Z. Sun, R. Albert, Node-independent elementary signaling modes: A measure of redundancy in Boolean signaling transduction networks, Network Science, 4 (2016) 273-292.
[61] J.C. Rozum, R. Albert, Identifying (un)controllable dynamical behavior with applications to biomolecular networks, bioRxiv, (2017).
[62] Z. Yuan, C. Zhao, Z. Di, W.-X. Wang, Y.-C. Lai, Exact controllability of complex networks, Nature communications, 4 (2013) 2447.
[63] Y.Y. Liu, J.J. Slotine, A.L. Barabasi, Controllability of complex networks, Nature, 473 (2011) 167-173.
[64] A. Mochizuki, B. Fiedler, G. Kurosawa, D. Saito, Dynamics and control at feedback vertex sets. II: A faithful monitor to determine the diversity of molecular activities in regulatory networks, Journal of theoretical biology, 335 (2013) 130-146.
[65] J.G.T. Zañudo, G. Yang, R. Albert, Structure-based control of complex networks with nonlinear dynamics, Proceedings of the National Academy of Sciences, 114 (2017) 7234-7239.
[66] G. Yang, J. Gomez Tejeda Zanudo, R. Albert, Target Control in Logical Models Using the Domain of Influence of Nodes, bioRxiv, (2018).
[67] A. K. Chandra, G. Markowsky, On the number of prime implicants, 1978.