Autonomous Modeling, Statistical Complexity and Semi-annealed Treatment of Boolean Networks

by

Xinwei Gong

Department of Physics
Duke University

Date: _____

Approved:

_____
Joshua E. S. Socolar, Supervisor

_____
Robert P. Behringer

_____
John Harer

_____
James Moody

_____
Christopher W. Walter

Dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in the Department of Physics
in the Graduate School of Duke University
2012

Abstract

Autonomous Modeling, Statistical Complexity and Semi-annealed Treatment of Boolean Networks

by

Xinwei Gong

Department of Physics
Duke University

Date: _____

Approved:

_____
Joshua E. S. Socolar, Supervisor

_____
Robert P. Behringer

_____
John Harer

_____
James Moody

_____
Christopher W. Walter

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Physics in the Graduate School of Duke University
2012

Abstract

This dissertation presents three studies on Boolean networks. Boolean networks are a class of mathematical systems consisting of interacting elements with binary state variables. Each element is a node with a Boolean logic gate, and the presence of interactions between any two nodes is represented by directed links. Boolean networks that implement the logic structures of real systems are studied as coarse-grained models of the real systems. Large random Boolean networks are studied with mean field approximations and used to provide a baseline of possible behaviors of large real systems. This dissertation presents one study of the former type, concerning the stable oscillation of a yeast cell-cycle oscillator, and two studies of the latter type, respectively concerning the statistical complexity of large random Boolean networks and an extension of traditional mean field techniques that accounts for the presence of short loops.

In the cell-cycle oscillator study, a novel autonomous update scheme is introduced to study the stability of oscillations in small networks. A motif that corrects pulse-growing perturbations and a motif that grows pulses are identified. A combination of the two motifs is capable of sustaining stable oscillations. Examining a Boolean model of the yeast cell-cycle oscillator using an autonomous update scheme yields evidence that it is endowed with such a combination.

Random Boolean networks are classified as ordered, critical or disordered based on their response to small perturbations. In the second study, random Boolean networks

are taken as prototypical cases for the evaluation of two measures of complexity based on a criterion for optimal statistical prediction. One measure, defined for homogeneous systems, does not distinguish between the static spatial inhomogeneity in the ordered phase and the dynamical inhomogeneity in the disordered phase. A modification in which complexities of individual nodes are calculated yields vanishing complexity values for networks in the ordered and critical phases and for highly disordered networks, peaking somewhere in the disordered phase. Individual nodes with high complexity have, on average, a larger influence on the system dynamics.

Lastly, a semi-annealed approximation that preserves the correlation between states at neighboring nodes is introduced to study a social game-inspired network model in which all links are bidirectional and all nodes have a self-input. The technique developed here is shown to yield accurate predictions of distribution of players' states, and accounts for some nontrivial collective behavior of game theoretic interest.

Contents

List of Tables

List of Figures

Acknowledgements

It has been a long and winding journey that has led to the completion of this thesis, and I am indebted to many people for their support along the way. I would like to first express my deep gratitude to my undergraduate advisors Profs. Michael Williams, Beate Schmittmann, and Royce Zia. They introduced me to the world of scientific research, challenged me with difficult problems, and kindled my desire for deeper explorations. My pursuit of a Ph.D. would not have started without them.

I would like to thank all the colleagues whom I have worked with over these last few years. Particularly, I sincerely thank Hans Norrell for providing academic assistance and personal encouragements. It was tremendously helpful to have a peer mentor in the same research group. I sincerely thank Volkan Sevim for spearheading the research project and putting in the great amount of effort that led to my first scientific publication. It was a project that I truly enjoyed working on, and it gave me much confidence in completing my Ph.D. studies. I also want to give special thanks to fellow group members Xianrui Cheng and Mengyang Sun for the camaraderie I have enjoyed over the years.

I thank all my committee members for their valuable guidance. Particularly I would like to thank Prof. Jim Moody for granting me the support of Duke Network Analysis Center and for fruitful discussions that made the end of my graduate career much smoother. I highly appreciate his trust in me to produce quality science and his willingness to be part of my dissertation committee with relatively short notice.

Most importantly, I thank my advisor Prof. Joshua Socolar. Josh is a laid-back yet demanding advisor. He is very respectful of students' personal time, considerate of their struggles, yet he always expects the highest level of rigor in their work. He has taught me to think meticulously carefully in the proposal, evaluation, and presentation of a problem. He has guided me through difficult questions and at the same time has given me the freedom and encouragement to explore different topics, find my own interest, and develop my own ideas. I am very grateful that I have had Josh as my mentor.

Finally, I would like to acknowledge friends and family near and far who have supported me all along. I thank my parents for their continued belief in me. I am grateful to be part of a close-knit class at Duke Physics, among which are Ivan, Horacio, Josh A. Josh K, Charles, Abhijit, Junyao, Wangzhi, Dong, Shangying, and Yingyi. I would like to thank other students in the physics department who have made the graduate school experience more enjoyable: Mauricio, Kristine, and Seth. During my time at Duke, I have been fortunate to have made great friends on campus and around the triangle area. With the risk of omitting many names, my gratitude goes to Reid, Mark, Ryan F, Ryan S, Jeff, Brian, Andrew, and especially Daniel and the Zeller family. All your love and support will be treasured.

1

Introduction

A complex system is a collection of interacting elements whose aggregate behavior and properties are not obviously derivable from the individual parts. Complex networks are a subset of complex systems in which elements and their interactions can be represented by a mathematical graph: each element is represented by a node, and the presence of the influence that one element has on another is represented by a directed link between the corresponding nodes. Elements in a complex network can have uniform or diverse individual properties, and interactions between elements can take the same or various forms. This loose definition of complex networks applies to systems studied in a wide array of scientific disciplines. For example, electronic circuits, gene regulatory networks, nervous systems, and economic markets can all be considered complex networks. Studying the mathematical models of complex networks may then elucidate the logical structure of these diverse types of real systems. The mathematical models usually require a combination of tools from nonlinear dynamics, statistical physics, probability theory and information theory. Physicists have traditionally utilized these tools to study various types of mathematical and physical systems, and in the last two decades have been increasingly active in the

study of complex networks [1, 2, 3].

The study of complex networks has two aspects: structure and state. The former concerns the connectivity of a network. The goals may include acquiring qualitative descriptions of connection patterns, understanding the dynamics of network formation, and building rewiring strategies that could maximize network performance. For example, Watts and Strogatz proposed an assembly scheme for "small-world networks" that mimics the high clustering and small characteristic path length usually seen in social networks [4]. Another example involves the observation that many real-world networks feature a small fraction of components connected to a large number of components while the majority have only sparse connections. The number of connections that each component has, or its degree, is further suggested to follow a power law distribution, also known as scale-free distribution, in many networks such as the world wide web, scientific coauthorship network, and network of sexual disease transmission. Several mathematical models have been successful in constructing scale-free networks using simple iterative procedures, *e.g.* [5, 6, 7, 8], the first and most famous one being growth with preferential attachment proposed by Barabási and Albert [9]. These two studies have led to an explosion of research and are exemplary cases of physicists leading the charge of network science.

This thesis, however, focuses on the second aspect of network studies: the dynamics of state variables associated with the network components. Such a variable can describe, for example, the expression of a gene in a regulatory network, the infection status of a person in a disease transmission network, or the excitation of a neuron in a nervous system. The detailed connections and interactions of these real systems are not usually known, so the theory of the dynamics on complex networks began with the simplest model rich enough to exhibit complex behaviors. Kauffman pioneered the effort by proposing large synchronous random Boolean networks (RBNs) as a coarse-grained model of genetic regulatory networks [10]. Each node is associated

with a binary state variable, determined by a quenched logic function drawn from a distribution. Kauffman found numerical evidence that RBNs can be categorized into ordered, disordered, and critical phases based on their number of attractors, attractor length and responses to small perturbations [11, 12]. This categorization was later given theoretical support [13, 14]. Kauffman further hypothesized that biological networks, which should be robust against perturbations, yet flexible enough to respond to some stimuli, belong to the critical phase.

The two aspects of complex networks, structure and state, are certainly not always decoupled. The dynamics of structure can affect the dynamics of state, and vice versa. Models of coevolution of structure and state are less common in the literature. One recent example, studied by Durrett *et al.*, is a voter network model in which connected individuals with different opinions may choose to agree, or may end their connection and seek other connections [15]. It is often appropriate to study the dynamics of state variables on a quenched network structure. For example, the structure of a gene regulatory network stays mostly the same over the lifetime of an organism and only changes considerably over evolutionary time scale. In the social network context, the number of individuals in a network and the relationship structure may change more frequently, but studying the state dynamics on a quenched structure could still be useful because it may illustrate how states change on a short time scale and elucidate the effect that certain topological features alone have on state dynamics. This thesis focuses on state dynamics on quenched network structures.

Kauffman's work inspired a large body of research into the properties and application of Boolean networks. The original research described in this thesis represents a part of that body of work. Two of the research projects are centered on random network models. One concerns the statistical complexity of the type of RBNs that was originally proposed by Kauffman. It is a study of the dynamics of RBNs

from a computational mechanics angle as well as a study of the complexity measures that aim to describe the amount of computational resource needed to simulate a physical process. Efforts on quantifying the complexity of extended systems have traditionally focused on spatially and logically homogeneous systems such as cellular automata [16, 17]. This work presents a new approach that quantifies the complexity of each component in a heterogeneous extended systems. This work has been published in *Physical Review E* [18].

The other study in this thesis concerning random networks diverges slightly from the traditional RBN models to include bidirectional links and self inputs on every node, which may represent the reciprocity and memory effects common in social interactions. The model simulates social networks in which a prisoners' dilemma game (PDG) is played between neighbors and players' strategies depend on their expectations of the reward. This social game-inspired model can be framed as a Boolean network in which each node's state is a function of its own state and the states of neighbors. Traditional mean-field analysis of RBNs based on an annealed approximation is not applicable on networks filled with short loops. A semi-annealed analysis is introduced to study the strategies and expected rewards for these PDG networks.

With the advent of high throughput genomic and proteomic analysis technologies, there has been more and more data available to guide the reconstruction of detailed structure and dynamical rules for gene regulatory networks. Subsequently there has been an increasing focus on local functions rather than global features of networks. Alon *et al.* found evidence that small sets of connected genes, or motifs, may be responsible for specific tasks as part of the overall functioning of a genetic network [19]. One research project described in this thesis contributes to that body of evidence. The work concerns how a yeast cell-cycle oscillator, or any small sets of nodes, can sustain stable oscillations. The traditional synchronous updating scheme

used for studying properties of large random networks lacks the neccessary detail for studying the dynamics of small networks. A novel Boolean modeling framework involving autonomous updating is introduced in this study and is shown to be capable of revealing the network motifs that make up a stable oscillator. This work was published in *PLoS Computational Biology* [20].

The structure of the thesis is as follows. Chapter 2 presents a review of some background knowledge about Boolean networks and information theory that is helpful for understanding the original research described in later chapters. Chapter 3 first presents an autonomous Boolean framework and a detailed study of the network structures that support stable oscillators under the autonomous model, followed by an investigation into whether the yeast cell-cycle oscillator is endowed with the oscillation sustaining structure. Chapter 4 presents the work on quantifying the statistical complexity of random Boolean networks. Chapter 5 concerns the semi-annealed treatment of random Boolean networks with short loops, and its application to a social-game inspired network model. A summary and some concluding statements are given in Chapter 6.

2

Background Knowledge

In this chapter, some basics of Boolean networks and information theory are reviewed. The discussion is divided into three sections. In Section 2.1, networks of interacting Boolean variables are introduced. These Boolean systems can evolve in discrete time steps or in continuous time. This section discusses the Boolean networks that updates in synchronous discrete time steps, which are the most basic Boolean network models. Large synchronous Boolean networks with random connections have been well studied in the past, and Section 2.1 reviews their properties such as different dynamical phases, steady-state bias, and sensitivity to small perturbations.

In Section 2.2, Boolean networks in which all variables do not update synchronously are considered. Many asynchronous Boolean network models exist [21, 22, 23], and the section focuses on the autonomous update scheme in which each interaction in the network is associated with a continuous time delay and a low pass filter clears out short pulses. The choice of the autonomous Boolean network model will be justified by a review of its advantages over traditional synchronous and continuous update models for the purpose of simulating real systems, paving the way for a discussion of the modeling of cyclic dynamics with autonomous Boolean network

model in Chapter 3.

Lastly, Section 2.3 reviews some information theoretic concepts, including Shannon entropy. These concepts provide the theoretical foundation for the complexity measure for spatially extended systems recently introduced by Shalizi *et al.* [17]. The complexity measures used in Chapter 4 are closely related to this measure.

2.1 Synchronous random Boolean Networks

Complex systems in which excitatory and inhibitory interactions occur among multiple elements are found in many physical, biological, and sociological contexts. Boolean networks are a class of models often used to gain insights into the dynamical properties of such systems in which interactions amongst individual agents may be approximated by binary logic functions. A paradigmatic case is the behavior of gene regulatory networks in which the expression of one gene may activate or repress the expression of others such that genes are turned "on" or "off" by the presence or absence of other genes. Boolean networks were first proposed by Stuart A. Kauffman in 1969 as a mathematical framework to study gene regulatory networks [10, 24]. At the time of their proposal, detailed genetic maps of any organisms were not available, and Kauffman used random network topologies to study the different qualitative behaviors exhibited by ensembles of random Boolean networks. The goal was to provide a baseline for studying the structure and dynamics of gene regulatory networks in the future. Random Boolean networks (RBNs) have since garnered much attention and become an active subfield of statistical physics. Features of RBNs such as steady-state bias, sensitivity to perturbations, attractor length and mutual information between nodes have been extensively investigated [13, 14, 25, 26]. Some of the main features are reviewed in this section.

Inputs \ Functions		ON	OFF	AND	OR	XOR	XNOR	NIF
0	0	1	0	0	0	0	1	0
0	1	1	0	0	1	1	0	0
1	0	1	0	0	1	1	0	1
1	1	1	0	1	1	0	1	0

Table 2.1: Examples of two-input Boolean logic functions

2.1.1 Definitions

A Boolean network consists of a number of nodes connected to one another through directed links. Each node is associated with a binary state variable and its value is determined by a Boolean logic function taking arguments from all nodes that have an output link connecting to the node. Most Boolean modeling efforts have focused on the synchronous update scheme, in which an external clock dictates the the timing of updates of all nodes. The mathematical representation of a synchronous Boolean networks is a set of Boolean equations:

$$x_i(t) = f_i(x_{i_1}(t-1), x_{i_2}(t-1), \ldots, x_{i_{k(i)}}(t-1)), \tag{2.1}$$

where t is a positive integer. Here x_i is the state of node i at time t, and it is either 1 (ON) or 0 (OFF); $f_i : \{0,1\}^n \rightarrow \{0,1\}$ is a quenched Boolean logic function for node i; i_j's are the quenched input nodes of node i, and $k(i)$ represents the number of input nodes that node i has. The choice of input nodes i_j to each node i defines the network structure, and the choice of functions f_i defines the dynamical rules. Some examples of two-input Boolean logic functions are shown in Tab. 2.1.

To initiate the network dynamics, one has to specify an initial condition, which is defined as the states of all nodes at $t = 0$: $\{x_i(0)\}$. Because each node can take one of two Boolean states, there are a total of 2^N possible network states, where N is the number of nodes in the network. Therefore the number of possible network states is finite for networks of finite size. Boolean logic functions are deterministic,

so each network state $\{x_i(t)\}$ corresponds to a unique $\{x_i(t+1)\}$. For each initial condition $x_i(0)$, the network updates at synchronous time steps and subsequently visits states $\{x_i(1)\}, \{x_i(2)\}, \{x_i(3)\}, \ldots$. Because there are only a finite number of total states, the network must eventually visit a state that it has been before, and because the update rules are deterministic, the network settles into a periodic attractor whenever it reaches previously visited state. Not every network state is part of a periodic attractor. A network can start on a state and reach an attractor that does not include the starting state. This type of state is called a transient state. A network can also start on a state and stay there, and such a state is termed a fixed-point. Note that fixed-point is a periodic attractor by itself.

In a random Boolean network, the Boolean functions are chosen randomly from some weighted distribution over the logic functions with the appropriate number of inputs, and the quenched choice of nodes i_j that act as inputs to i are randomly selected, independently for each i, from the full set of nodes, with each given equal weight. The number of inputs and outputs per node, which may or may not be the same for all nodes, and the distribution of logic functions are the parameters that characterize an ensemble of synchronous RBNs. Kauffman focused on networks where all nodes have the same number of inputs, and that is also the class of networks Chapters 4 and 5 of this thesis focuses on.

2.1.2 Bias and sensitivity

An ensemble of RBNs is a collection of networks that have the same distributions of numbers of input/output links and of logic functions. The study of ensembles of RBNs provides a baseline for the study of more specific Boolean networks. For example, when an interesting feature in a particular real network system is observed, one can ask whether that feature can be explained by a random network with similar network parameters. Statistics on large random networks of finite size can be approx-

imated by studying random networks of size in the thermodynamic limit $N \to \infty$, and quantities of interest are calculated as ensemble averages.

One measure that has been well-studied is the *bias* ρ, which is defined as the fraction of nodes in the network with state 1:

$$\rho(t) \equiv \frac{1}{N}\sum_{i=1}^{N} x_i(t). \tag{2.2}$$

After transients have decayed, the bias can settle down to a steady value, or it can jump between a number of values, having a cycle of its own. Because the network state is after all periodic, the bias cycle has to be periodic as well. This thesis is mostly concerned with cases where the bias reaches a stable fixed-point. This is the most prevalent behavior in sparse networks near criticality (define below). Additionally, transcriptional data from yeast and other organisms show that the gene regulatory functions most likely belong to a class of functions that stabilizes the bias, making the case of stable bias the most biological relevant one [27, 28].

The bias of a network ensemble can be obtained by a mean-field technique called the *annealed approximation*. First introduced in [13], the annealed approximation assumes that the logic functions and inputs of each node are reassigned at each time step, according to the desired connectivity and function distributions. This is very different from the quenched structure and logic of the Boolean network defined in 2.1. The stochasticity introduced by the reassignment breaks the periodic cycles and any state correlations between nodes. However, in the thermodynamic limit, the correlations between nodes become negligible, and the bias of an annealed network approximates the bias of a quenched network with the same distribution of links and Boolean functions.

Reference [28] presents a method of calculating the bias in the annealed approximation using the bias map for networks in which all nodes have the same number of

inputs. A simple extension to accommodate networks with a distribution of numbers of inputs is found in [14]. Here the former case is presented, where each node has k inputs. The evolution of the network bias is represented by the bias map:

$$\rho(t+1) = g(\rho(t))\,. \tag{2.3}$$

To obtain an expression for $g(\rho)$, the annealed approximation assumes no state correlations in the network and lets every node have an equal likelihood ρ of being ON when the network bias is ρ. Then, the probability of the K inputs of node i forming the sequence $\mathbf{x} = x_{i_1}, x_{i_2}, ..., x_{i_K}$ is

$$\Pr(\mathbf{x}|\rho) = \rho^{|\mathbf{x}|}(1-\rho)^{K-|\mathbf{x}|}\,, \tag{2.4}$$

where $|\mathbf{x}|$ is the number of 1s in $\mathbf{x}$. The probability of node i being ON at the next time step is then $\sum_{\mathbf{x}} f_i(\mathbf{x})\Pr(\mathbf{x}|\rho)$. Subsequently, averaging over all nodes, one can obtain the bias map as

$$g(\rho) = \left\langle \sum_{\mathbf{x}} f_i(\mathbf{x})\Pr(\mathbf{x}|\rho) \right\rangle_i\,, \tag{2.5}$$

where $\langle \bullet \rangle_i$ denotes averaging over all nodes, effectively meaning taking the expectation over the distribution of logic functions. In other words, the evolution of the bias follows the following equation:

$$\rho(t+1) = \left\langle \sum_{\mathbf{x}} f_i(\mathbf{x})\rho(t)^{|\mathbf{x}|}(1-\rho(t))^{K-|\mathbf{x}|} \right\rangle_i\,. \tag{2.6}$$

Equation 2.6 may have nontrivial solutions, such as 2-cycle solutions, depending on the chosen Boolean function distribution. However, it is expected that all biologically realistic Boolean function distribution would finally lead the bias into a unique stable fixed point ρ^* [28], and this steady-state bias is obtained by solving for ρ^* in $\rho^* = g(\rho^*)$. Note $\sum_{\mathbf{x}} \Pr(\mathbf{x}|\rho) = 1$ and $f_i(\mathbf{x}) \in \{0,1\}$, so one can obtain $g(\rho) \in [0,1]$.

Note that the bias is an ensemble measure. It gives the average of fraction of nodes that are ON for a network in the ensemble, averaged over initial conditions. Two ensembles can have the same bias but very different types of dynamics in its networks. For example, a $K = 2$ ensemble with all XOR functions has the same bias as a $K = 2$ ensemble with half of the functions as ON and half of them as OFF. Both ensembles have bias 0.5, but in the former ensemble, nodes are constantly switching between states 0 and 1, and in the second ensemble, a node's state is either constantly 0 or constantly 1. It is then clear that another measure is needed to characterize the dynamics more fully.

Before introducing this measure, another quantity, Hamming distance, needs to be introduced. Consider two copies of the same network in states A and B, and denote the states of the nodes $\{x_i^A\}$ and $\{x_i^B\}$ respectively. The Hamming distance is the number of nodes whose states differ between A and B:

$$H^{AB}(t) = \sum_{i=1}^{N} \left| x_i^A(t) - x_i^B \right| . \tag{2.7}$$

The Hamming distance is a measure of how different two states are.

If two network realizations start out with states that are different at only 1 node, then the expected Hamming distance at the next time step gives a measure of how sensitive a network is to perturbations. This sensitivity measure, denoted λ, is analogous to the Lyaponov exponent that measures the rate of divergence of two close orbits in continuous systems. Networks with $\lambda < 1$ tend to correct perturbations, and multiple copies of the same network initiated at slightly different states converge to the same trajectory. Networks with $\lambda > 1$ would let perturbations grow and multiple networks initiated with slightly different states would reach different attractors. Networks with $\lambda = 1$ tend to sustain the magnitude of perturbations, so multiple networks initiated with slightly different states could reach the same attractor but

out of phase. The sensitivity λ is given by the following expression:

$$\lambda = \left\langle \sum_{i=1}^{K} \sum_{\mathbf{x}} (f(\mathbf{x}^{(i,0)}) \oplus f(\mathbf{x}^{(i,1)})) \rho^{|\mathbf{x}|} (1-\rho)^{K-|\mathbf{x}|} \right\rangle_i , \tag{2.8}$$

where $\mathbf{x}^{(i,j)} = (x_1, ..., x_{i-1}, j, x_{i+1}, ..., x_K)$ and $\oplus$ denotes the XOR function [14]. The three qualitatively different network behaviors distinguished by sensitivity are termed ordered ($\lambda < 1$), disordered ($\lambda > 1$), and critical ($\lambda = 1$) [29].

2.2 Autonomous Boolean Networks

Boolean networks, when used in the study of real physical systems, are idealized representations of the underlying continuous and stochastic processes. Continuous network models, such as sets of differential equations, are traditionally employed to study these processes. The Boolean abstraction is adopted in the hope that its simplicity can help elucidate the underlying mechanisms of the real systems where insights are usually hard to obtain with continuous models. The formalism of the Boolean network model introduced in Eq. 2.1 dictates that all nodes update synchronously at discrete time steps. It has been demonstrated that the synchronous updating scheme introduces a large number of attractors not found in several classes of continuous systems [30, 31, 32]. A natural modification to the synchronous Boolean model is to introduce continuous time delays while keeping the simplicity of the Boolean state space.

Boolean delay equations (BDEs), first introduced by Dee and Ghil in [33], are a type of autonomous Boolean network model that can be described mathematically by the following formulas:

$$y_{ij}(t) = x_j(t - \tau_{ij}) \tag{2.9a}$$

$$x_i(t) = f_i(y_{i1}(t), y_{i2}(t), ..., y_{in}(t)), \quad t \geqslant 0 \tag{2.9b}$$

These two equations state that at time t, the state x_i is determined by "memorization variables" y_{i1}, y_{i2}, ..., y_{in}, which are the respective states of nodes 1, 2, ..., n at different earlier times $t-\tau_{i1}$, $t-\tau_{i2}$, ..., $t-\tau_{in}$. In other words, τ_{ij} is the time delay between the instant when node j updates and the instant when that update has an effect on node i. One can picture that in this model whenever a node switches state, it sends a signal to each one of its downstream nodes, informing them of its new state, and the subsequent state switches of the downstream nodes are their reactions to receiving those signals.

The space of initial conditions for this update scheme is different from that of the synchronous Boolean model. In order for a time series of all nodes to be uniquely specified, it is no longer sufficient to specify all state values at a given instance, but now it is necessary to specify the time series of all nodes for a certain duration of time. For example, determining the time series of a node i starting at $t = 0$ requires the knowledge of all the signals that started traveling to it from each of its upstream node i_j since $t = -\tau_{ij}$. Thus, an initial condition for Eq. 2.9 is defined to be the time series of all nodes in the interval $[-\max\{\tau_{i1}, \tau_{i2}, ..., \tau_{in}\}, 0)$.

Dee, Ghil and Mullhaupt have uncovered many properties of BDE systems [33, 34, 35]. Even though a BDE model eliminates the artificial synchrony of the traditional Boolean model, it is found that if all time delays are rational, then a BDE system can be reduced in effect to synchronous Boolean networks, in which case the dynamics would be eventually periodic. Aperiodic behavior can arise if there are incommensurate time delays in the system.

A simple system for demonstrating the origin and difference in these behaviors involves one node with two self inputs that make an XOR gate and a constant value in its initial history:

$$x(t) = x(t-\tau_1) \oplus x(t-\tau_2) \quad t \geqslant 0; \qquad x(t) = 1, \quad t < 0, \tag{2.10}$$

where $\oplus$ denotes an XOR gate. The first two signals are sent at $t = 0$ when the x switches from 1 to 0 because of the initial history. If the time delays are rational, say $\tau_1 = 1$ and $\tau_2 = 1.1$, then one can insert 9 and 10 nodes each with a simple copy function to the two links respectively to make two loops of lengths 10 and 11. Also let each link in this new network have time delay 0.1. Then the network becomes a synchronous network with 0.1 as a single time step. The state variable x has the same behavior on the original BDE system and the equivalent synchronous system. Such a reduction to a synchronous network is however not possible if τ_1 and τ_2 are irrationally related, say when $\tau_1 = 1$ and $\tau_2 = \sqrt{2}$. A node with XOR logic has the property that whenever it receives a new signal, its state switches. In the current case, the original two signals would reach the node at different times, creating subsequent switching events that lead to more and more signals that will not be commensurate. In other words, no two signals will reach the node at the same time. Thus, the number of switching events per unit time can become arbitrarily large while pulse width tend towards 0. It has been observed in some networks that the frequency of switching events grows as power law in time, creating an "ultraviolet catastrophe." Even in the former case where the BDE system can be reduced to a synchronous network and eventually settles into periodic dynamics, switching events typically happen on the time scale of the new reduced step length in the equivalent synchronous network and thus the average time span between switches for a node can be shorter than a typical time delay. [34, 35].

Note that in the former case where the time delays are rational, if the time delays had been $\tau_1 = 1$ and $\tau_2 = 1.01$, then the equivalent synchronous network would have time step 0.01, in which case the average number of state switches per unit time grows another 10 fold compared to when $\tau_2 = 1.1$. The closer τ_2 approaches 1, then more frequently switching events occur. However, in the case where $\tau_1 = 1$ and $\tau_2 = 1$ when the network is synchronous to start with, the two original signals arrive at the same

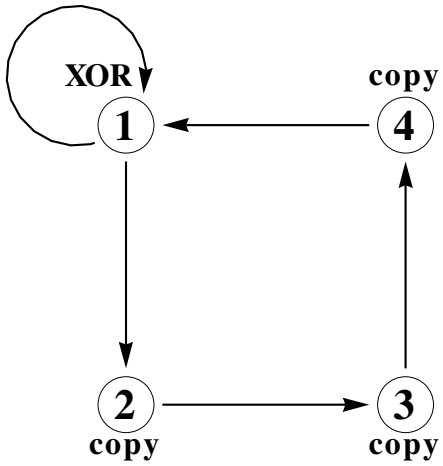

FIGURE 2.1: A network with 3 copiers and an XOR gate

time to keep $x = 0$ at $t = 1$, and the systems goes into a fixed-point attractor with no more switching events. Thus, changing one time delay in synchronous Boolean network $x(t) = x(t-1) \oplus x(t-1)$ from 1 to $1 + \delta$ (for arbitrarily small δ) changes the system's long term behavior drastically. Such a synchronous network, in the view of the BDE framework, is therefore "structurally unstable." Another example of this effect is seen by comparing the time series of the network shown in Fig. 2.1, computed using different update schemes. A typical time series of the network under the synchronous model is shown in Fig. 2.2(a), a typical time series under the BDE model of almost synchronous time delays is shown in Fig. 2.2(b).

In the context of modeling real systems, arbitrarily rapid switching of state is often impossible. For example, Zhang *et al.* studied a type of electronic logic gate whose minimum pulse width is around 0.2 nanoseconds [36]. Hasty *et al.* studied the chemical reactions involved in an autoregulation pathway of bacteriophage λ and found the expression time limit to be on the order of seconds [37]. The dense pulses that a BDE network typically creates thus make it an nonideal framework to study real systems.

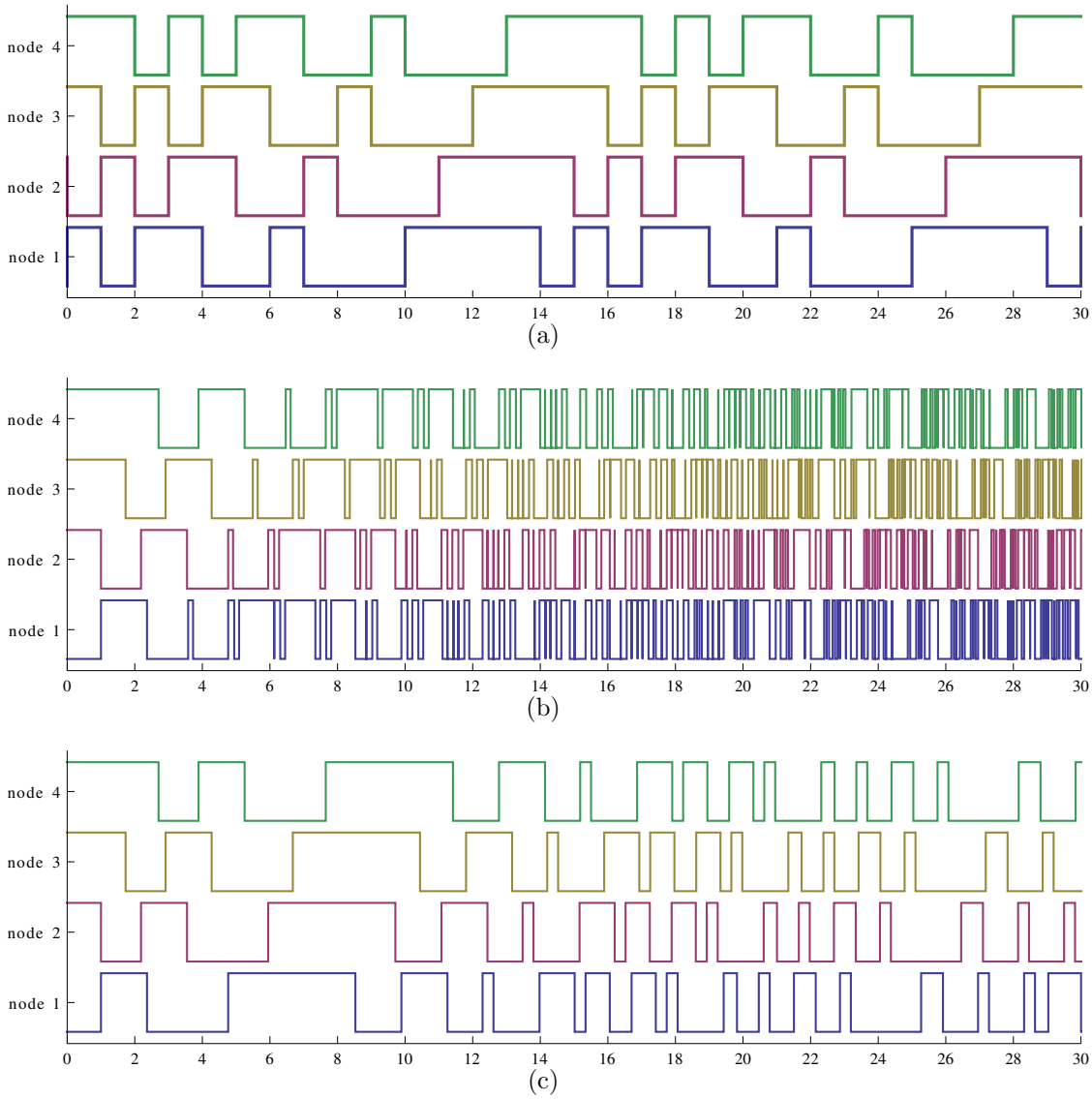

FIGURE 2.2: Typical time series of the network in Fig. 2.1 under the update scheme of (a) synchronous network, (b) Boolean delay equations, and (c) Autonomous updates short pules rejection.

Synchronous networks can be structurally unstable and BDE networks often give rise to unwanted short spikes. A new modeling framework that preserves the structural stability of the BDE model while eliminating short spikes is desired. One natural way to achieve this is to introduce a short-pulse rejection mechanism to the BDE framework to fix the accumulation of fast switching events: two consecutive

switching events on the same node that happen within a short time τ_s are deleted from the time series and neither event would have any bearing on the downstream nodes. In other words, a low-pass filter is adopted to suppress spikes of duration much shorter than a typical time delay. Klemm and Bornholdt introduced an equation of evolution aimed at eliminating short pulses in networks with almost synchronous updates [38]:

$$x_i(t) = \Theta\left[(2\tau_s)^{-1} \int_{t-\tau_s}^{t+\tau_s} f_i(y_{i1}(t'), y_{i2}(t'), ..., y_{in}(t'))dt' - \frac{1}{2}\right], \qquad (2.11)$$

where $\tau_s \ll \tau_{ij}$, and $\Theta(h)$ is the Heaviside step function; $\Theta(h) = 0$ for $h \leqslant 0$ and 1 for $h \geqslant 0$. Let us briefly check what this equation means. The first term inside the bracket, $(2\tau_s)^{-1}$ times an integral, is the fraction of time in the window $(t-\tau_s, t+\tau_s)$ that x_i would be ON if short pulses in x_i were not rejected. If this fraction is larger than 1/2, then x_i is deemed to be ON at t by the current autonomous model, and it is considered OFF otherwise. This averaging process can certainly eliminate a short pulse in the window $(t-\tau_s, t+\tau_s)$ if it is the only short pulse present. However, if multiple short pulses are found in the same window, the averaging process could actually introduce shorter pulses. Klemm and Bornholdt introduced Eq. 2.11 to filter out short pulses in synchronous networks where only one timing perturbation is applied at a time. For a BDE network though, where typically all time delays are different and short pulses aggregate in a dense window, another short-pulse rejection mechanism is needed. Such a mechanism is described in Chapter 3.

Figure 2.2(c) shows a typical time series of the network in Fig. 2.1 under autonomous updating with short pulse rejection where pulses shorter than $\tau_s = 0.2$ are eliminated. Note that the three time series shown in Figs. 2.2(a), 2.2(b) and 2.2(c) come from the network topology and are generated with really close time delays. Only autonomous updating with short pulse rejection produces realistic time series,

free of forced synchronization and narrow spikes. In Chapter 3, this modeling framework will be further evaluated, expanded with the introduction of stochastic time delays, and utilized in the analysis of a biological system.

2.3 Theoretical foundations of complexity measures

This section takes a detour from the dynamics of Boolean networks and focuses on another aspect in the study of dynamical systems – their complexity.

In information theory, entropy is a measure of the uncertainty, or unpredictability, associated with a random variable. For example, tossing of a fair coin has positive entropy, because there is no way to be certain of the outcome. On the other hand, tossing of a coin with two heads is a process with zero entropy because you always know what is the next outcome. Shannon was the first person to quantify the expected amount of information stored in a random variable [39]. To understand the quantification of information, first consider the following motivations. For a random variable X that produces numerical outcomes of $a_1, a_2, \ldots, a_n$, with probabilities $p_1, p_2, \ldots, p_n$ respectively. A measure of information should satisfy the following conditions:

1. The information content of an output a_i is not related to the numerical value of a_i, but is related to its probability of occurring p_i. This information content is denoted $I(p_i)$, and it is usually called the *self-information* or the surprisal of the output a_i;

2. Self-information $I(p_i)$ is a continuous function of p_i;

3. Self-information $I(p_i)$ is a decreasing function of p_i. That is, the more likely the outcome a_i is to occur, the less surprised one is to see the outcome, and thus the less self-information the outcome possesses.

4. If $p_i = p_{i,1} \times p_{i,2}$, then $I(p_i) = I(p_{i,1}) + I(p_{i,2})$. That is, if event i is the composite of two statistically independent events, then the self-information of event i should be the sum of the self-information of the two separate events.

Only logarithmic functions satisfy these properties, and thus self-information of an outcome a_i may be written as

$$I(p_i) = -\log(p_i)\,. \tag{2.12}$$

Therefore, the expected amount of information revealed by the outcome of a particular source (such as a random variable) is the weighted average of the various outputs:

$$H[X] = \sum_i p_i I(p_i) = -\sum_i \Pr(X = x_i) \log \Pr(X = x_i)\,, \tag{2.13}$$

where the minus sign is introduced to keep the self-information positive. This quantity is usually referred to as the Shannon entropy. The common value for the logarithmic base is 2, and when base 2 is used, the unit for Shannon entropy is the `bit`.

The Shannon entropy measures the average amount of information in a variable. It relates to complexity of a process, say $\{x(1), x(2), \ldots, x(t)\}$, because complexity has historically been linked to the entropy of the output of the process. In this case, complexity only measures the level of randomness of the output. The complexity of a process has also been associated with the entropy of the state transitions, for example the expected self-information of a step $x(t) \to x(t+1)$ in the process. More generally, for a non-Markovian process, the self-information of a larger step may be used: $\{\ldots, x(t-2), x(t-1), x(t)\} \;\; \to \; \{x(t+1), x(t+2), x(t+3), \ldots\}$. Such complexity measures, however, only captures the randomness of the process. For example, throwing a fair coin has a higher process entropy than, say, organized flipping of a coin from one side to another every time step.

It was suggested that complexity should not be conflated with randomness, and at the same time completely uniform processes should be not considered complex either [40, 41, 42]. Complex and interesting behavior should lie somewhere in the middle, and a measure of complexity should capture not only the statistical properties but also describe the how information is stored and processed.

Grassberger, Crutchfield and Young came up with such a measure for processes producing a single output variable, and defined complexity as the entropy of *causal states* in the network [43]. A causal state is a collection of states that corresponds to the same distribution of future states. The Grassberger-Crutchfield-Young statistical complexity is then the least amount of information about the past trajectory required for optimal prediction of future trajectory. This measure is calculated from time series data alone, without reference to the physical laws that govern the dynamical processes. Shalizi et al. extended the concept to processes with spatial extent [17]. The basic ideas are summarized here. For details and information–theoretic support of these ideas, see Ref. [44].

Given a field X that varies over space and time in a system where information propagates at a maximum speed of c, the past light cone of a space-time point $(\vec{r}, t)$ consists of all space-time points where events can influence $X(\vec{r}, t)$. The future light cone consists of all points that can be influenced by $X(\vec{r}, t)$. $L^-(\vec{r}, t)$ and $L^+(\vec{r}, t)$ denote the configurations of the field X in the past and future light cones, respectively:

$$\begin{aligned} L^{\pm}(\vec{r}, t) &= \{(X(\vec{s}, u), |t-u|, \vec{r}-\vec{s}) \\ &\qquad \forall\ u \gtrless t \text{ and } |\vec{s}-\vec{r}| \leqslant c|t-u|\}. \end{aligned} \tag{2.14}$$

Note that each element of $L^{\pm}$ consists of three quantities: a field value, a time lapse, and a *relative* position. Thus L^- (or L^+) is exactly the same for two distinct space-time points with identical past (or future) light cones.

Each space-time point is associated with one L^- and one L^+. An ensemble of such

pairs defines a probability distribution $P(L^+|L^-)$ for future light cone configurations conditioned on past configurations. A causal state $\epsilon(L^-)$ is defined as a set of past light cone configurations that have the same distribution of future configurations. All instances of a given causal state predict the same distribution of future light cone configurations:

$$\epsilon(l^-) = \{\omega : P(L^+|L^- = \omega) = P(L^+|L^- = l^-)\}. \tag{2.15}$$

By definition, $\epsilon(l^-)$ is a sufficient local statistic; knowing the causal state at a given point provides the same predictive power as knowing the exact past light cone configuration for the purpose of predicting future dynamics. $\epsilon(l^-)$ is also a *minimal* sufficient statistic [45], meaning that the sufficient statistic $\epsilon(l^-)$ contains the least amount of information among all statistics that have the same predictive power:

$$H[\epsilon(l^-)] \leqslant H[\eta(l^-)], \tag{2.16}$$

where $\eta(l^-)$ is a sufficient statistic and $H[X] = -\sum_i P(X = x_i) \log_2 P(X = x_i)$ denotes Shannon entropy. It then follows that $H[\epsilon(l^-)]$ is the least amount of information for optimal prediction of the future dynamics [44, 17], which is taken to be the relevant measure of a system's complexity. We use the shorthand notation

$$C \equiv H[\epsilon(l^-)]. \tag{2.17}$$

The value of C depends upon the choice of the ensemble of space-time points used to determine the causal states. In the work of Shalizi *et al.*, which concerns the complexity of homogeneous systems, causal states are determined at any given time by considering the ensemble of spatial locations in the system at that time. In other words, for this complexity measure, denoted C_μ, causal states $\epsilon_\mu(l^-,\ t)$ are defined as in Eq. 2.15 with the restriction that $P(L^+|L^-)$ is determined from the set of past and future light cone pairs present at time t. This approach allows one to speak of the

complexity of a system as a function of time, which may exhibit transient dynamics. Systems that exhibit a spontaneous increase in $C_\mu(t)$ have been described by Shalizi *et al.* as going through a self-organization process [17].

For analyzing heterogeneous systems, a different choice of the ensemble of space-time points used to determined the causal states may be more appropriate. A detailed definition and analysis of the complexity of random Boolean networks is presented in Chapter 4.

3

Autonomous Boolean Modeling of Oscillation-stabilizing Networks

Complex systems with interconnected components may exhibit properties that are not obvious from the properties of individual components or interactions. These complex systems are found in many physical, biological, and sociological contexts. A paradigmatic case which the present chapter focuses on is a regulatory network in which genes activate or repress the expression of one another to perform an enormous number of operations required for normal functioning of a cell. The development of high-throughput genomic and proteomic technologies in recent years has enabled biologists to conduct large-scale surveys of gene expression for different organisms and infer information on individual gene interactions [46, 47]. As a result, major progress has been seen in constructions of detailed genetic networks of certain well-studied organisms. Knowledge of such a network, in turn, sets the stage for explaining cellular functions in terms of variations in gene expression. Due to the complicated nature of such systems, mathematical models play an important role in elucidating their emergent properties.

Mathematical models of regulatory networks vary in their levels of description. A common class of models express the regulatory interactions in terms of ordinary differential equations and associates each gene with a number of kinetic constants and parameters. Such a model introduces the difficult tasks of exploring a large parameter space and extracting meaningful information from a large set of continuous time series. A simpler model system that has received much attention from the biology community is the Boolean network model [10]. In a Boolean network model, gene expression is quantized in only two states 1 (ON) and 0 (OFF), and the state of each gene is determined by a Boolean logic function that depends on the states of a set of other genes. It has been shown that meaningful biological information such as temporal order of expression and cell type can be extracted from gene expression data entirely in the binary domain [48]. The Boolean approximation can give a clear representation of the order of gene activation and is thus more likely to provide intuition about qualitative features of a network such as motifs that support stable oscillations. A Boolean model is adopted in this investigation. Continuous models may be used in the future to confirm the generality of the conclusions and to investigate more detailed features of the network.

To show that the ordered sequences of activation follow periodic patterns in a Boolean model, one needs to pay attention to the timing of update events. Most of the Boolean modeling efforts have focused on the synchronous update scheme where an external clock dictates the synchronous updates of all nodes according to their local logic functions at each time step. There are known artifacts of this update scheme such as false synchrony and the production of marginally stable attractors [30, 38], as noted in Section 2.2. Furthermore, different interactions in a regulatory network are associated with different biological processes, and thus different time delays. A more realistic Boolean model should take into account this timing differentiation. An autonomous update scheme is adopted here. This scheme assigns a unique time

delay to each link of the network, and it allows the implementation of infinitesimal timing perturbations, which is employed to determine stability of attractors. Section 3.1 introduces this autonomous model in detail and give a description for its computational implementation. Section 3.2 presents an investigation of small network dynamics under the autonomous Boolean framework, with emphasis on the stability of oscillatory behavior. Two local functional structures, or "motifs," that can sustain stable oscillations in small networks. This insight allows us to investigate if a real system known to support stable oscillations is utilizing similar mechanisms as these motifs.

One system of interest is a recently proposed yeast cell-cycle oscillator. Cell-cycle, the series of events that take place in a cell that lead to its division and replication, is a fundamental process to living systems. Expression levels of many genes exhibit oscillatory behavior over a series of cell cycles, resembling a periodic attractor in dynamical systems. Understanding how a regulatory network responsible for the cell cycle operates is of great interest to biological scientists. In Section 3.3, this yeast cell-cycle oscillator is examined with the autonomous update scheme, and evidence is found that it is endowed with a combination of the two motifs that support stable oscillations.

This work was spearheaded by a former postdoctoral fellow Volkan Sevim of the research group headed by Joshua E. S. Socolar. My (the author's) role in the work included developing an algorithm and a computer program to simulate autonomous Boolean networks with noisy time delays and short-pulse rejection, incorporating the yeast cell-cycle oscillator into simulations, designing a subroutine that recognizes cyclic behavior, running a small portion of the simulations, and writing the computer program to visualize Boolean series. I also participated in discussions with Sevim and Socolar on the analyses of network structure that support stable oscillations and the yeast cell-cycle oscillator.

3.1 Implementing the autonomous Boolean network model

We have discussed in Section 2.2 the formalism for an autonomous update scheme, Boolean delay equations. The model can be summarized by these two equations:

$$y_{ij}(t) = x_j(t - \tau_{ij}) \tag{3.1a}$$

$$x_i(t) = f_i(y_{i1}(t'), y_{i2}(t'), ..., y_{in}(t')). \tag{3.1b}$$

Here y_{ij}'s are memorization variables that represent the states that nodes observe of each other due to the effect of time delays τ_{ij}'s. We also noted that this model introduces the accumulation of short pulses that are not found in real systems, and that a method for eliminating short pulses in almost synchronous networks does not work in a general autonomous framework. A new short-pulse rejection mechanism is thus desired. From the perspective of modeling real dynamical systems, it is natural to introduce noise to the continuous time delays. While Klemm and Bornholdt have applied single small timing perturbations to almost synchronous networks to test the reliability of attractors found in the synchronous Boolean model [38], a full-fledged noisy autonomous network model with short pulse rejection was not introduced until this work. There are mathematical and computational issues to work out in implementing continuous noisy time delays on a Boolean network. The presence of short-pulse rejection only complicates the matter. Working out these issues and implementing the model are my main contributions to the research project.

First of all, simulating autonomous updating is very different from simulating synchronous networks. In the latter case, a simple computer program that keeps a time step count and updates each node at every time step would suffice. An event-driven algorithm to simulate the continuous updates is developed here. In brief, the algorithm can be described using a picture in which each node sends a signal to its downstream nodes to inform them of its new state whenever it experiences an

update. When a node receives a signal, it checks for whether its own state would be switched by the signal. Our program utilizes a global queue that stores all these signal arrival events, each containing the information of the nodes and times involved, in chronological order of the arrival time. At each iteration in our program, the first entry in the queue is examined; if it triggers an update, then subsequent signals that it causes are added to the queue, and the system time is reset to the time of this update.

One important note to make on this event driven strategy concerns the initialization of the program. As discussed in Section 2.2, the initial condition in this time delay system should be the collection of all nodes' switching events in $[-\max\{\tau_{ij}\}, 0)$. These events send signals to their downstream nodes which may arrive before or after $t = 0$. Because the network is supposed to start operating at $t = 0$, the signals that arrive before this time are ignored and not included in the queue. Furthermore, even though there may not be any signal arrivals in the global queue at the instant $t = 0$, every node still needs to be checked at $t = 0$ for possible updating against the initial history of its upstream nodes, because the model dictates that the time-delayed logic is consistent starting this moment.

The short-pulse rejection mechanism embedded in Eq. 3.1b presents a hurdle in developing the program. Under short-pulse rejection, whenever a node switches state, it sends signals to its downstream nodes, but that switching event may then be deemed to never have happened if an opposite switching event on the same node happens subsequently within a time span of τ_s, the short-pulse rejection parameter. The signals that are already sent have to be taken back and deleted, so the model requires that these signals cannot have already effected any further state changes within the duration, effectively placing a constraint on the relationship between the shortest time delay and τ_s: $\tau_s < \min\{\tau_{ij}\}$. Note that τ_s would be further restricted after the introduction of noisy time delays. Short-pulse rejections are achieved with

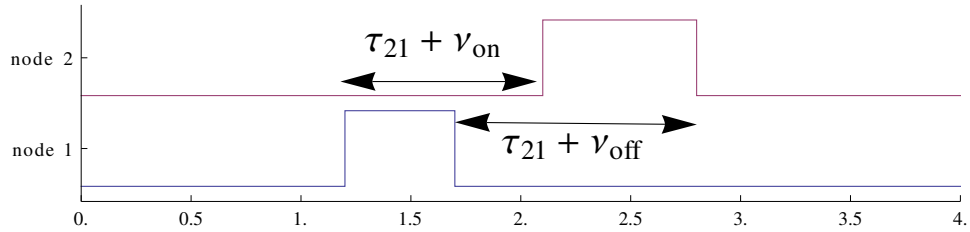

FIGURE 3.1: Two adjacent switching events can have different delays due to timing perturbations

a method that essentially holds every update for a duration τ_s before deeming it a real update. If an opposite switching event is reached by the program probe during this duration, then both events are deleted. Otherwise the original event and the signals it creates are added to their respective places in the event queue as soon as the holding period ends.

Another challenge is to implement timing perturbations. To describe perturbations of time delays mathematically, a noise term is added in the definition of memorization variables:

$$y_{ij}(t) = x_j(t - \tau_{ij} + \nu(t)), \tag{3.2}$$

where $\nu(t)$ is a random variable drawn from a uniform distribution on $[-\nu^{max},\ \nu^{max}]$ with $0 < \nu^{max} \ll \tau_{ij}$. The physical root of the noise term lies in the fact that update signals don't always take exactly the same time to travel across the same link. Once a signal reaches a downstream node, it would take effect and that effect should not be temporarily reversed, as a continuously varying $\nu(t)$ may cause. A way to simulate the noise term that better conforms to the physical processes is to generate a new value of ν every time a signal starts traveling on a link. The sum of this value and the quenched time delay τ on the link becomes the new time it takes the signal to travel across the link, as demonstrated by Fig. 3.1. The addition of small perturbations in the time delay parameters allows the study of stability of attractors in the Boolean context.

We note that this noise simulating strategy effectively renders Eq. 3.2 inexact, because $\nu(t)$ is not well defined except for instants when signals arrive, and it also makes possible an undesirable scenario in simulations of the network dynamics that needs to be addressed: two signals traveling on a same link can switch order if the noise term associated with the later signal is much smaller than that of the earlier one. This can qualitatively alter local dynamics. For example, the transmission of an ON-OFF pulse from node j to node i should leave i in an OFF state, but i would be left in an ON state if the OFF signal arrives before the ON signal. This big change is not innate to the equations of time evolutions, but it is an artifact of the strategy in noise simulation. There are two ways to fix this artifact. A restriction can be placed on the noise level: $\nu^{max} \leqslant \tau_s/2$, so that two consecutive signals on a link never switch order. In the more general case without that restriction, two signals are deleted whenever they cross each other. One can picture that when the trailing edge of a pulse catches up with the leading edge, the pulse annihilates and neither signal has any effect on the downstream node. An illustration is given in Fig. 3.2. This "zero-pulse annihilation" mechanism can be thought as a special case of short-pulse rejection that does not happen on a node, but within a link.

We note that even without noisy time delays, an autonomous Boolean model can allow two consecutive signals to have different time delays. This is a plausible modeling strategy because two consecutive signals on the same link have to be a ON signal and a OFF signal, and in real networks, ON and OFF switching events are associated with different physical processes and affect their downstream targets at different times. In such cases, the zero-pulse annihilation mechanism needs to be implemented to avoid the crossing of signals.

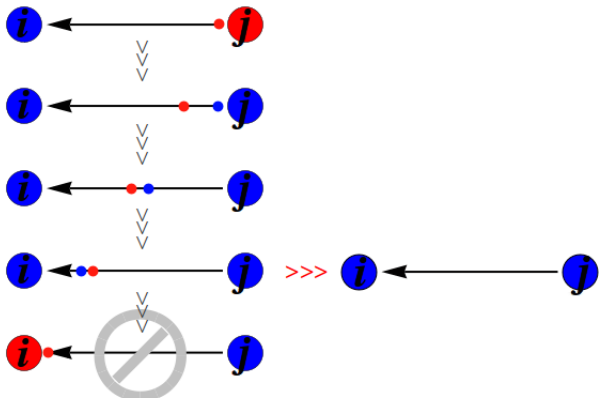

FIGURE 3.2: A schematics of pulse annihilation. Node j, which has an input to node j, first turns ON (represented by color red) and then OFF (color blue). The two switching events generates two signals on the link between j and i. The two signals make the leading the trailing edges of a pulse. When differences in τ_{on} and τ_{off} or noise causes the leading and trailing edges of an pulse to get closer and closer, the trailing edge may surpass the leading edge, eventually leaving i constitutively ON. This behavior is not only unrealistic for real physical systems, but is also inconsistent with the original autonomous model assumptions. A zero-pulse annihilation mechanism is implemented to rectify this artifact: whenever a pulse's width becomes zero during its path along an edge, it is deleted.

3.2 Cyclic dynamics on the noisy autonomous Boolean network

Periodic behavior is prevalent in biological systems. To achieve periodic oscillations in a network model, the autonomous Boolean model defined in Section 3.1 is studied here with a focus of the dynamics on simple loops. One simple loop with 4 copiers is shown in Fig. 3.3(a). This network has two fixed-point attractors: all ON and all OFF, independent of the update scheme. With synchronous updating, where the initial condition is a snapshot of the network's state, every initial state would be revisited 4 time steps later, if not earlier, so there are no transients in this case and every one of the 16 possible states is part of an attractor. For autonomous updating without noise, all update events in the initial history are reproduced and recycled around the loop (barring short pulses), so oscillation is a common behavior. Also, once timing perturbations are introduced, each pulse would shrink or grow when it

travels from one link to another. In this fashion, noise causes the pulse width to execute a random walk when it travels around the loop. In the long time limit, it would eventually become short enough or long enough for the short-pulse rejection or the zero-pulse annihilation mechanisms would eliminate it or the short dip created by it. All pulses eventually die out and the network settles into one of its fixed-point attractors (Fig. 3.3(b)). Thus, any oscillating state is only marginally stable on this four-node loop, and this is true for simple loops of copiers of any length. In a loop with an even number of NOT gates, if ones redefines the states between any two NOT gates to be their opposite, the resultant network is a simple loop of copiers. Thus, any simple loop with an even number of NOT gates cannot sustain stable oscillations.

Oscillations could be stable, however, if an odd number of the nodes in the simple loop are NOT gates. One example is shown in Fig. 3.4(a). There is no fixed-point attractor for this logic configuration, so with both synchronous updating and autonomous updating, all attractors have oscillations. Figure 3.4(b) shows a typical time series of the network under autonomous updating. When there is no noise, short pulses from the initial history propagates around the loop, similar to simple loop with no noise, but now the pulses are inverted every time they arrive at the NOT gate. When timing perturbations are introduced into the network, the short pulses are again eventually eliminated, and the network dynamics becomes a single kink propagation: each switching event in the time series is directly caused by the one preceding it. Neither noise nor any other effect can stop the oscillation because no stationary state can satisfy all four logic functions simultaneously. The network is thus "forced" to oscillate. Such a network is called a "frustration oscillator" [20]. Negative feedbacks like the repressing link in Fig 3.4(a) are common sources of frustration [49, 50, 51, 52]. In the regulatory network context, gene 2 is repressed by the expression of 1, but when 1 is not expressed, 2 must be able to build up sponta-

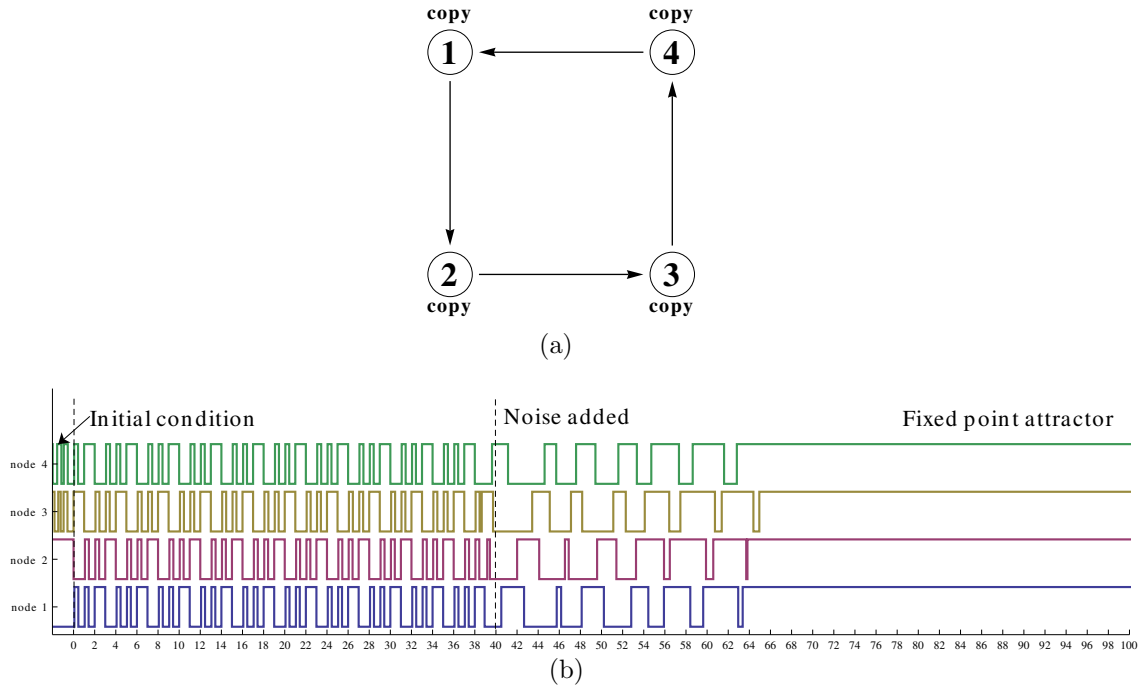

FIGURE 3.3: (a) A simple loop of 4 copiers and (b) a typical time series of the loop under noisy time delays. This simple loop has two fixed-point attractors, all ON and all OFF. Because of short-pulse rejection and zero-pulse annihilation, all pulses and dips would eventually disappear under noisy time delays, and the loop would settle into one of the two fixed-points. No oscillation can be sustained on a simple loop in the limit of $t \to \infty$.

neously, implying the presence of a constitutive input or a positive autoregulation.

Are constitutively expressed genes or the frustration mechanism required for stable oscillations in regulatory networks? It is shown next that in the Boolean network context, it is possible for oscillations to be stable on the backbone of a simple ring like the one in Fig. 3.3(a), without the help of frustratied logic, when certain extra components are added to the ring. The simple network structures, or motifs, that sustain stable oscillations can be constructed by combining a pulse-rectifying motif and a pulse-growing motif, both introduced below.

The function of a pulse-rectifying motif is to correct pulse-growing perturbations. A simple example of it is constructed by adding an auto-repressing link to a node

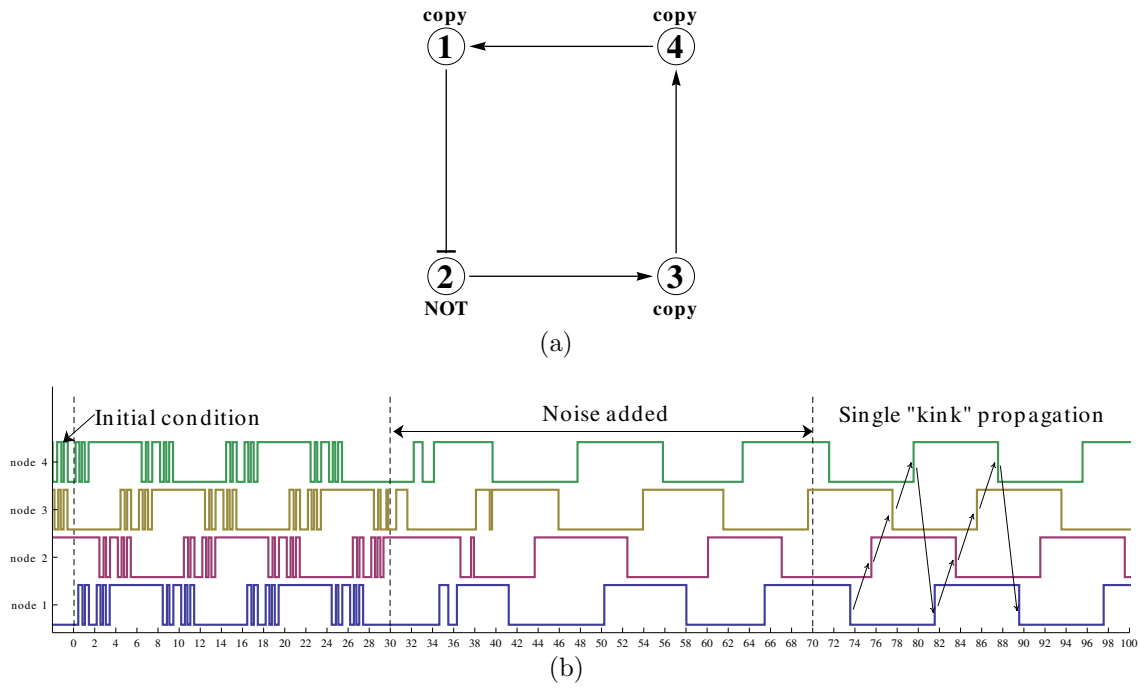

FIGURE 3.4: (a) A frustration loop with 3 copiers and a NOT gate, and (b) a typical time series of the loop under noisy time delays. This loop is called a frustration loop because it has no fixed-point attractors and is thus "forced" into oscillation. After noise has smoothed out short pulses, only dynamics left is single kink propagations. That is, each updated is triggered by the one immediately preceding it. Any pulse where the trailing and leading edges do not have a causal relation cannot sustained in a frustration oscillator in the limit of $t \to \infty$.

in a ring of copiers to make that node a NIF gate. Figure 3.5(a) shows such an example. The NIF gate makes the repressing link dominant, meaning that if node 1 gets a ON single from itself, it has to turn OFF, no matter what state it sees node 4 is in. So everytime a pulse larger than τ_{11} passes through node 1, it would be cut back to a width of τ_{11}. In the presence of noise, the pulse width at node 1 is bounded by an upper limit of $\tau_{11} + \nu^{max}$. Any pulse that is significantly long would be cut off at node 1 to a width within $\tau_{11} - \nu^{max}$, $\tau_{11} + \nu^{max}$. If a pulse that is within this range experiences a net growth as it travels around the loop, it would be cut back to within the range as it passes node 1. Figure 3.5(b) shows a single

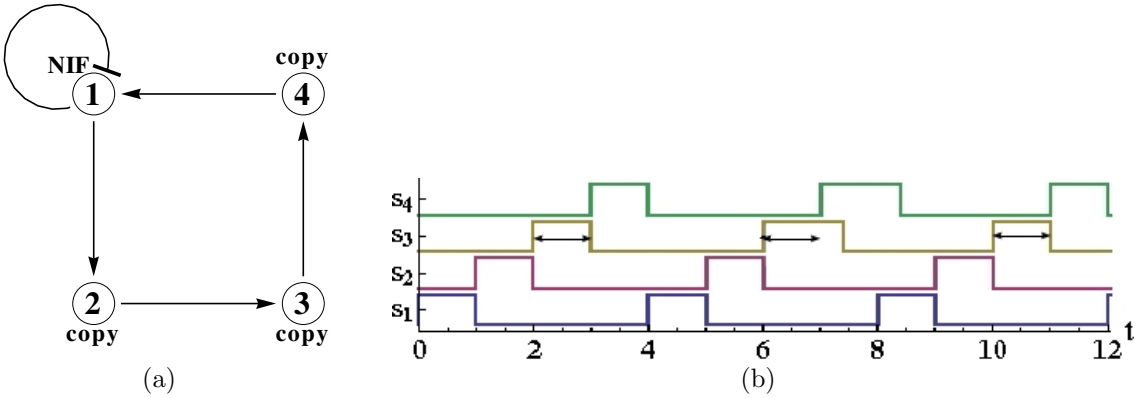

FIGURE 3.5: (a) a pulse-rectifying motif that corrects pulse-growing perturbations; (b) a time series showing a pulse-growing timing perturbation being corrected by the self-repressing link one cycle after the perturbation is introduced.

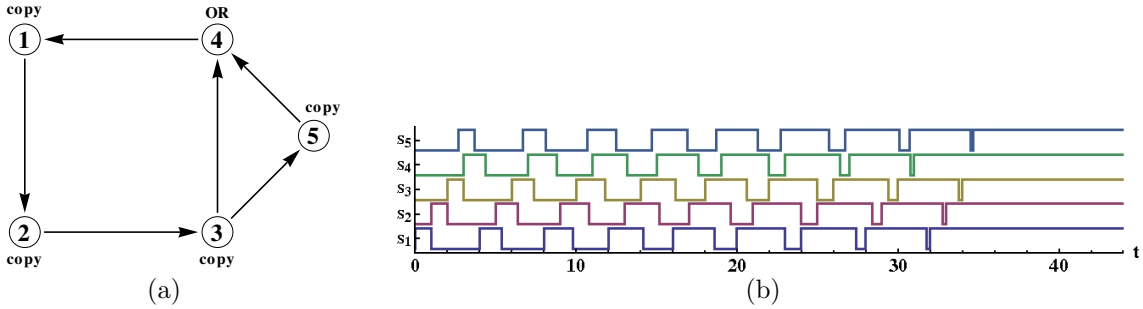

FIGURE 3.6: (a) a motif that grows pulses perturbations by letting them travel on two alternative routes before converging at an OR gate; (b) a time series showing the steady growth of pulse width in the absence of timing perturbations.

pulse-growing timing perturbation being corrected by the NIF gate soon after it is introduced. This rectifying motif, however, can do nothing to correct pulse-shrinking perturbations which are also bound to be present where there is noise. Another type of motif that serves to grow pulses, however, can counteract these pulse-shrinking perturbations.

To construct a pulse growing motif, one can simply add an extra link between two connected nodes, or, in the more biologically relevant case, another path with extra nodes between two connected nodes, on a simple ring of copiers (Fig. 3.6(a)). The node with two inputs would be an OR gate. The additional components create an

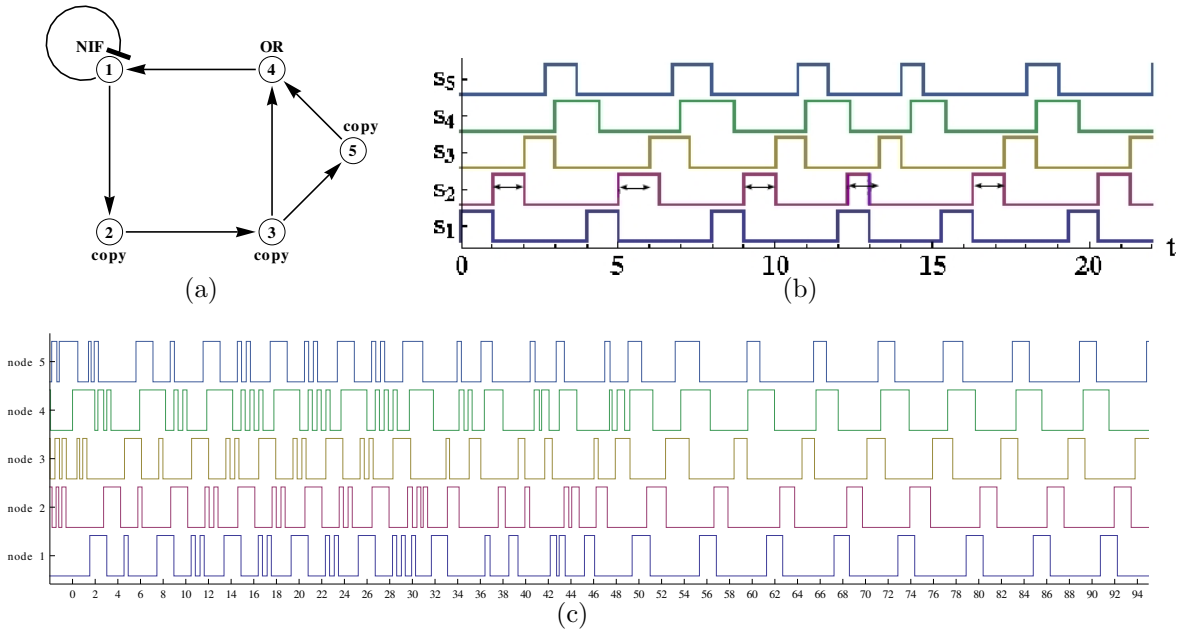

FIGURE 3.7: (a) a combination of a rectifying motif and a growing motif that is capable of sustaining stable oscillations; (b) a time series showing both the one-time pulse shrinking and pulse-growing perturbations are corrected after one cycle; (c) a typical time series with a random initial condition. After short pulses are filtered out, the network dynamics becomes the stable transmission of a single pulse.

alternative pathway for pulses to travel, between nodes 3 and 4 in this case. The two pathways have time delays τ_{43} and $\tau_{45} + \tau_{53}$ respectively. In general, these two time delays are different. If node 3 experiences a pulse of width τ, where $\tau > |\tau_{45} + \tau_{53} > \tau_{43}|$, then the pulse would be grown by an amount of $|\tau_{45} + \tau_{53} > \tau_{43}|$ when it reaches node 4.

Now two motifs have been identified: a motif that rectifies pulse-growing perturbations and another motif that grows pulses. The two motifs can be combined to create a network that stabilizes oscillations on the simple ring backbone (Fig. 3.7(a)): one motif to grow a pulse as it travels around the loop, counteracting any pulse-shrinking perturbations, and the other to cut the pulse width back to a defined range if it is grown too wide. As is shown in Fig. 3.7(b), both the one-time pulse shrinking and pulse-growing perturbations are corrected after one cycle. Figure 3.7(c) shows

a typical time series with a random initial history under noisy autonomous updates. Noise filters out short pulses after a certain time, and the network dynamics settles into the propagation of a single pulse. Such a network does have a fixed point, the all-OFF state, and the stable oscillations it sustains are not supported by a frustration mechanism. Instead, it is an oscillator that supports the stable transmission of a traveling pulse. It can be termed a "transmission oscillator."

3.3 The yeast cell-cycle oscillator

The combination of motifs introduced in the last section can be the conceptual core for a network sustaining stable oscillations, but to show that an oscillator actually utilizes such a combination can be a non-trivial task. The extra components added to a simple ring to makes it a pulse-rectifying or growing motif can come in various forms, so their combinations can be hard to recognize. In this section, the subject of interest is a network that has been suggested to sustain oscillations in the gene regulatory context. In the autonomous Boolean network context, the question of whether this network could make a stable transmission oscillator with the right set of logic functions and time delays is explored.

Two decades ago, studies of the gene regulatory network of yeast, one of the most studied organism to date, revealed that two classes of regulatory molecules, cyclins and cyclin-dependent kinases (CDKs), determined a cell's progress through the cell cycle [53]. However, recent studies have demonstrated that in yeast cells where the functions of cyclins or CDKs are suppressed, a significant fraction of periodic genes are still expressed periodically [54, 55]. These studies give evidence for an independent transcriptional cell-cycle oscillator in budding yeast. Orlando *et al.* proposed such a oscillator in 2008 [55], shown in Fig. 3.8. The regulatory functions are not completely known. In [55], the oscillator was studied with the synchronous Boolean model and 8 "biologically interpretable" logic configurations (Tab. 3.1).

Besides the all-OFF fixed point, all 8 logic configurations are capable of producing at least two out of the three periodic attractors that match the experimentally observed expression order. However, it is an open question whether this match is caused by artifacts of the synchronous update scheme. None of the logic configurations makes the oscillator exhibit the frustration mechanism. Several alternative pathways and negative feedbacks in the oscillator are reminiscent of the pulse-growing and rectifying motifs, suggesting that this oscillator may contain a combination of motifs that supports a periodic attractor. One possibility is that the two loops consisting of CLN3, SBF, SFF, and ACE2 or SWI5 (one in each loop) are the backbone of transmission oscillations and that all other nodes in the oscillator are part of the extra components added to these simple rings to make for functioning oscillation-stabilizing motifs.

To test the hypothesis that a combination of growers and rectifiers is responsible for the stable oscillations obtained with the yeast oscillator, the network dynamics is simulated with a large set of time delays for each logic configuration. Then, realizations that lead to stable oscillations are identified, and possible rectifying links on these realizations are disabled to test if only pulse growth occurs. Specifically in exploring network dynamics, the initial condition of the network is defined to be all nodes OFF except a positive pulse at CLN3 between t=0 and 1.5. Time delays are chosen from a uniform distribution on [0.5, 2]; perturbation on time delays are uniformly drawn from [-0.05, 0.05]; and the short-pulse rejection threshold is $s = 0.1$. Network dynamics is simulated with each set of logic configurations 10000 times, with a different set of time delays each time, and each simulation runs up to 125000 updates. Noise is turned on between the 800^{th} and 90000^{th} updates to eliminate marginally stable oscillations. Another 10000 updates are run after the noise is turned off, and then the network behavior is observed. It should be noted that the choice of initial condition is rooted in the experimental observation that

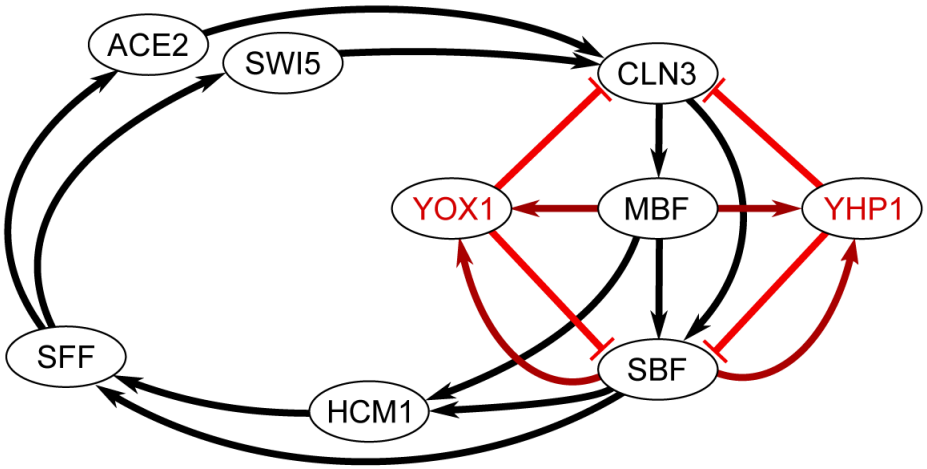

FIGURE 3.8: A recently proposed yeast cell-cycle oscillator

Transcriptional Factor	Boolean Function
MBF	CLN3
SBF	(CLN3 or MBF) AND NOT (YOX1 f_1 YHP1)
YOX1	MBF AND SBF
HCM1	MBF AND SBF
YHP1	MBF OR SBF
SFF	SBF f_2 HCM1
ACE2	SFF
SWI5	SFF
CLN3	(SWI5 f_3 ACE2) AND NOT (YOX1 YHP1)

Table 3.1: Regulatory logic functions for the yeast cell-cycle oscillator. Functions f_1, f_2 and f_3 can each be an AND or an OR

cell-cycle sequence starts when the cell commits to division by activating CLN3 [56]. Only one initial condition is chosen because the goal of this report is not to test the robustness of any attractors against variations in initial conditions, but just to investigate the mechanisms supporting stable oscillations. Another initial condition, turning CLN3 ON at $t = 0$ and leaving it ON until a repressing signal turns it OFF, yields very similar statistics.

Three types of long-time dynamical behavior are identified: the all OFF fixed

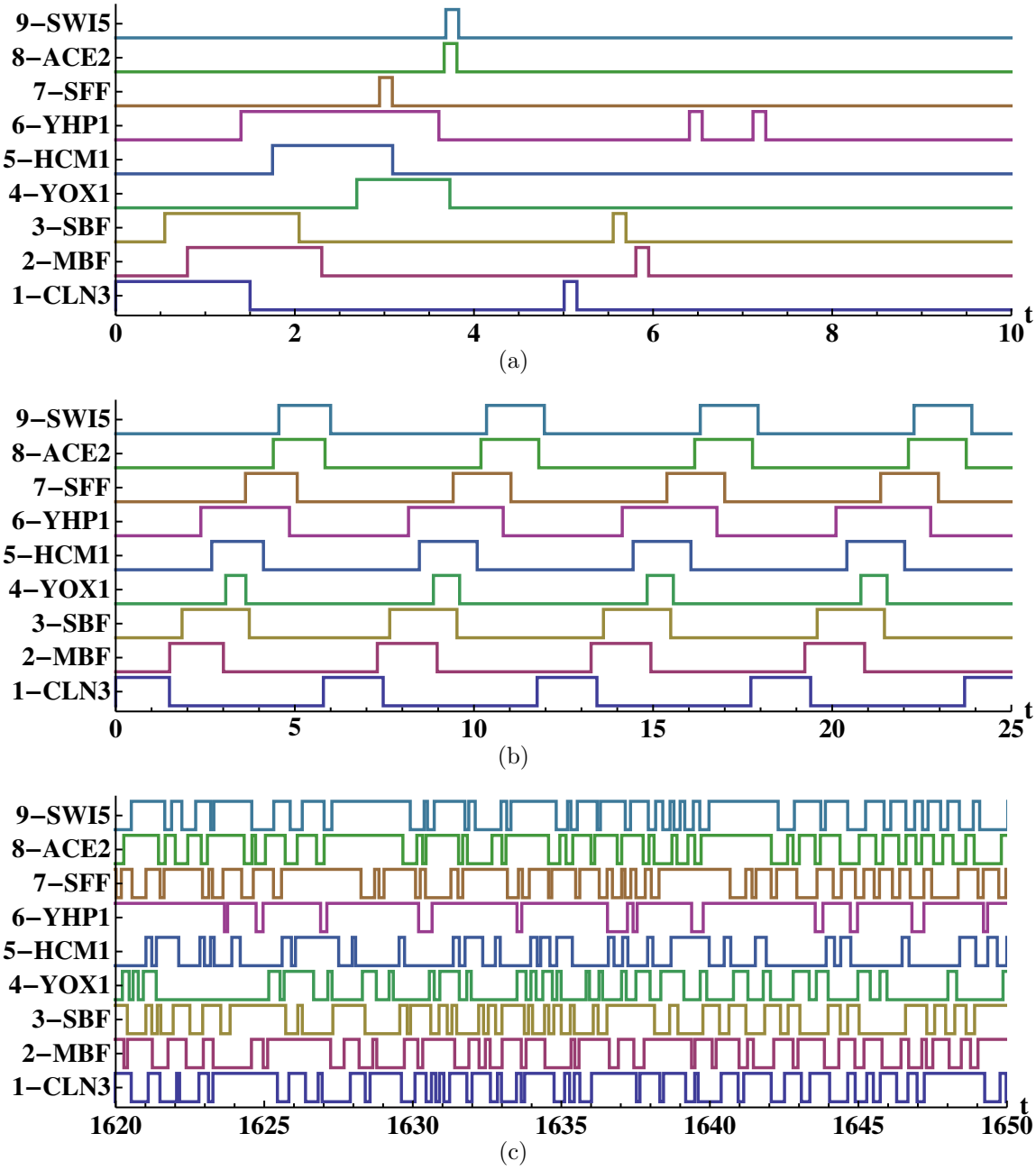

FIGURE 3.9: Three typical time series of the autonomous Boolean network model the yeast cell-cycle oscillator: (a) all nodes OFF; (b) periodic attractor; (c) complex oscillations that do not appear to be periodic.

point, stable periodic oscillations, and complex oscillations that appear aperiodic in the time scale observed (Fig. 3.9). Periodic oscillations can have periods that contain only a single pulse or multiple pulses. Only the periodic single-pulse oscillations are expected to be biologically relevant, and it is on networks realizations that yields these single-pulse oscillations that the investigation into whether they contain a grower-rectifier combination is carried out. The fraction of realizations that results in oscillating behaviors are shown for each logic configuration in Tab. 3.2. The numbers are consistent with the assumption that embedded rectifier and growers exist. For example, when the logic function f_2 or f_3, both cooperative logic between two activators, is an OR, SFF or CLN3 is most likely part of a pulse-growing motif, and when f_2 or f_3 is an AND, SFF or CLN3 is expected to be part of a pulse-shrinking motif. As can be seen in Tab. 3.2, if two sets of logic configurations are only different by f_2 or f_3, the OR function always yields a larger fraction of oscillating attractors than the AND function. If both f_2 and f_3 are an AND function, pulse-shrinking should be the dominant theme in the network dynamics and the network would reach an all-OFF fixed point. Statistics shown for Config. #1 and #5 confirms these expectations.

YOX1 and YHP1 are the only repressors in the network, and they only act as repressors, so they should be part of a rectifying motif that corrects pulse growth for the single-pulse attractors. Any rectifying sub-motifs should contain at least one of two repressors. Two relatively obvious possible rectifying routes are (a) CLN3 $\rightarrow$ MBF $\rightarrow$ YHP1 $\dashv$ CLN3 and (b) SBF $\rightarrow$ YHP1 $\dashv$ SBF, but there may also exist other routes of rectification. It is hard to identify the particular locus of growth or rectification in a typical network realization, the subgraphs that act as the growing or rectifying motifs can be intertwined and can be different for different realizations of the same logic configuration. Without attempts to identify the particular growing and rectifying motifs for each network realization, their presence is tested with the

Config. #	f_1	f_2	f_3	Single-pulse oscillations	Other oscillations
1	AND	AND	AND	0	0
2	AND	AND	OR	0.13	0.05
3	AND	OR	AND	0.3	0.22
4	AND	OR	OR	0.35	0.65
5	OR	AND	AND	0	0
6	OR	AND	OR	0.04	0.01
7	OR	OR	AND	0.14	0.05
8	OR	OR	OR	0.3	0.58

Table 3.2: Fraction of 10000 network realizations that results in single-pulse oscillations and all other types of oscillations for each of 8 logic configurations associated with the 3 unspecified functions in Tab. 3.1. Standard deviations are $O(10^{-2})$ given a bin size of 100.

following reasoning and procedure.

If the two only repressors YOX1 and YHP1 are eliminated from the network realizations that yield single-pulse oscillations, pulse growth should proceed cycle after cycle without interruption, and the network would eventually settle into an all-ON fixed point. This expectation is tested by first selecting the network realizations that are found to sustain single-pulse oscillations with the previous numerical method, disabling YOX1 and YHP1 after a network has settled into a stable single-pulse oscillation, and then observing the consequent behavior. Indeed, all these network realizations go to the all-ON fixed point. An example is shown in Fig. 3.10. This collapse to the all-ON fixed point is not caused by noise. In most cases, convergence onto the all-ON attractor is rapid. An initial pulse with width equaling 1.5 time units traveling on a simple loop backbone, whose total time delay averages at $4 \times 1.25 = 5$ time units, is much more likely to reach the all-OFF fixed point than the all-ON fixed point, considering only noise. Thus, numerical evidence supports the hypothesis that a pulse-growing motif and a pulse-rectifying motif are responsible for the stable oscillation seen in the yeast cell-cycle oscillator.

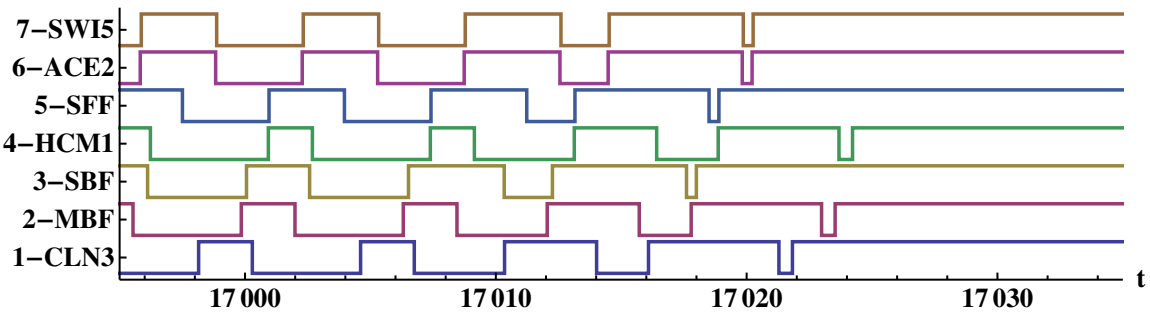

FIGURE 3.10: All nodes in the yeast cell-cycle oscillators turns ON soon after the two repressors are disabled.

3.4 Conclusion

The Boolean network model described here implements autonomous updating, a short-pulse rejection mechanism, and noisy time delays, and a zero-pulse annihilation mechanism. The model fixes the artifacts of some traditional Boolean models and permits stability analysis through the introduction of stochastic effects. In this autonomous Boolean framework, simple motifs that can sustain stable oscillations are identified, each of which consists of a pulse-growing sub-motif and a rectifying sub-motif. A numerical study that explores 8 sets of possible logic configurations and a large set of time delay parameters on a recently proposed yeast cell-cycle regulatory network gives evidence to the claim that stable oscillations on this network are supported by a grower-rectifier combination. The analysis also lends strong numerical support to the claim that the yeast cell-cycle oscillator is a transmission oscillator, in contrast to most biological oscillators seen in the literature, which are frustration oscillators. Continuous models may be employed in the future to confirm that the mechanisms discovered here in the Boolean context are also found in continuous state space.

The autonomous Boolean network model employed here is a novel mathematical model that can be applied to study a large number of systems. The insights it

generates regarding the yeast cell-cycle oscillator may be just a hint to its wider applicability. We further note that there are many interesting mathematical questions that arise from this model. For example, what are the attractor lengths, structures, and numbers in small networks, and how are the dynamics of large networks related to that of synchronous networks in different dynamical phases? The continuous time delays allow the possibility of Boolean chaos, so some natural questions to ask are how to measure the presence of Boolean chaos, and what are the criteria a network should satisfy to generate chaotic Boolean series. These mathematical questions are in turn related to physical problems such as generating random numbers with electronic circuits and the proposed critical behavior of large gene regulatory networks. Some of these questions have been examined by our group and our colleagues since the completion of this work [57, 36, 58].

4

Quantifying the Complexity of Random Boolean Networks

Large random Boolean networks (RBNs) were first studied by Kauffman as random models of gene regulatory systems [10, 24]. Since then, a large body of work has been done to understand their dynamical properties. We have previously critized that syncrhonous Boolean networks are not ideal models for studying real physical systems because they introduce artificial synchronicity and produce a large number of marginally stable attractors. However, because they are a class of complex networks whose dynamics is well understood, they make good model systems in studies related to quantifying the notion of complexity.

In computational mechanics, the complexity of a process that generates a single time series is defined as the least amount of information required for a maximally accurate statistical description of the series [59, 41, 43]. This definition classifies random processes, as well as simple periodic ones, as having low complexity. In a 2004 article, Shalizi et al. extended the definition to spatially–extended dynamical systems and introduced an algorithm for measuring the complexity of a discrete system given time series data for all components [17]. Applying the algorithm to

2D cyclic cellular automata (CCA) confirmed that it classified CCA rules generating fixed states or incoherent local oscillations as having low complexity and cases that produce turbulent spiral waves as having high complexity.

Shalizi's complexity measure, C_μ, is defined as the amount of information stored in local causal states, where a causal state is an equivalence class of all past configurations that give rise to the same distribution of future outcomes. A set of causal states can be discerned from time series data for all elements in the system. C_μ is obtained by measuring correlations between truncated past and future light cones associated with each element in the system.

The formalism developed by Shalizi et al. is intriguing due to its principled application of information theoretic concepts to the analysis of complex dynamical systems. It is not immediately clear, however, how the formalism can be applied to random or heterogeneous networks. We show here that a natural extension requires distinguishing between averages over nodes and averages over time, and also that the relevant quantities can be computed analytically for certain classes of random Boolean networks. Our primary goal is to clarify the meaning of Shalizi's complexity measure. In doing so, we propose a new measure of the information processing occurring at a given node, which gives a new statistic for distinguishing the roles played by different nodes in a complex network. We show that this measure depends upon global features of the dynamics and, in particular, that it is not simply related to the sensitivity of the network and is not maximized in critical networks. Along the way, we also present a conjecture concerning the dynamics of Boolean networks consisting only of parity functions.

We study two complexity measures of RBNs that differ only in the choice of the ensembles of spacetime points used for averaging the local complexity. One approach considers the ensemble of all spatial points at the same time instant, which corresponds to Shalizi's C_μ, previously discussed in Section 2.3. In computing C_μ, all

nodes are given equal weight in determining the probabilities of observing different states. Shalizi et al. used C_μ to investigate self-organization of cellular automata, which are logically and topologically uniform networks of discrete, interacting elements.

A second approach is to assign a complexity to each individual element by averaging over time. The system average of these individual complexities is denoted C_ν. For a homogeneous system in which all elements have statistically indistinguishable time series (a regular lattice with rules that lead to turbulent dynamics, for example), C_μ and C_ν are the same. We suggest that C_ν is the one that is most informative for spatially inhomogeneous systems.

To develop conceptual insights into the behavior of the quantities C_μ and C_ν, we study their dependence on the parameters specifying different ensembles of random Boolean networks (RBNs). RBNs have garnered much attention in the last few decades, and features such as steady-state bias, sensitivity to perturbations, attractor length and mutual information between nodes have been extensively investigated [13, 14, 25, 26]. We show here that for any given distribution of logic functions, the value of C_μ for a RBN can be analytically calculated as a function of the bias ρ and is not simply related to the sensitivity λ. C_ν on the other hand, is always near zero for sensitivity values $\lambda \leqslant 1$, where the network is in the ordered or critical regime, and is also zero for the highest possible λ value, where the network dynamics is strongly chaotic. Thus C_ν reflects the intuitive notion that systems with short periodic cycles or apparently random behavior should both have low complexity. The maximum of C_ν for RBNs occurs somewhere in the disordered regime. We find also that the amount of information processed by any given individual depends on global properties of the network dynamics as well as the logic functions of the node in question and others in its neighborhood. Along the way, we also present a conjecture concerning the dynamics of Boolean networks consisting only of parity functions.

Originial research presented in this chapter is my own work, performed under the guidance of Joshua E. S. Socolar.

4.1 Complexity of Random Boolean Networks

We study $C_\mu(t)$ and $C_\nu(r)$ in synchronously updated RBNs to determine how (or whether) they are related to well-understood measures of the dynamics, such as the sensitivity of the network or the overall bias in the values of the binary variables. In a synchronous RBN, at each time step each node i is updated based on a logic function f_i that is applied to the current values of its input nodes:

$$x_i(t) = f_i(x_{i_1}(t-1), x_{i_2}(t-1), \ldots, x_{i_n}(t-1)), \tag{4.1}$$

where t is a positive integer. Here $f_i : \{0,1\}^n \to \{0,1\}$ is a quenched Boolean logic function for node i, chosen randomly from some a weighted distribution over the logic functions with the appropriate number of inputs, and the quenched choice of nodes i_j that act as inputs to i are randomly selected, independently for each i, from the full set of nodes, with each given equal weight. The x_{i_j}'s are the binary values of the inputs to i. An important feature of RBN's is that the density of feedback or feedforward loops of any finite size goes to zero as the network size goes to infinity [31]; the local structure around each node is "tree-like."

The number of inputs and outputs per node, which may or may not be the same for all nodes, and the distribution of logic functions are the parameters that characterize an ensemble of synchronous RBNs. From these parameters, two global measures, the bias and the sensitivity, can be calculated analytically [13, 14]. The bias ρ is the fraction of nodes with value 1 after transients have decayed. Let $\mathbf{x} \in \{0,1\}^K$ be an input vector of length K, and K be the fixed number of input per node

for a network, then the bias map of the network is given by

$$\rho(t+1) = \left\langle \sum_{\mathbf{x}} f(\mathbf{x}) \rho(t)^{|\mathbf{x}|} (1-\rho(t))^{K-|\mathbf{x}|} \right\rangle, \tag{4.2}$$

where $\langle \bullet \rangle$ denotes expectation taken over distribution of logic functions and $|\mathbf{x}|$ is the number of 1s in $\mathbf{x}$. The fixed point or cyclic behavior of the bias is determined by the bias map. The sensitivity λ is the average rate of increase of the Hamming distance between two state space trajectories that initially differ at a single node:

$$\lambda = \left\langle \sum_{i=1}^{K} \sum_{\mathbf{x}} (f(\mathbf{x}^{(i,0)}) \oplus f(\mathbf{x}^{(i,1)})) \rho^{|\mathbf{x}|} (1-\rho)^{K-|\mathbf{x}|} \right\rangle, \tag{4.3}$$

where $\mathbf{x}^{(i,j)} = (x_1, ..., x_{i-1}, j, x_{i+1}, ..., x_K)$ and $\oplus$ denotes the XOR function [14]. The sensitivity distinguishes qualitatively different network behaviors that have been termed ordered ($\lambda < 1$), disordered ($\lambda > 1$), and critical ($\lambda = 1$) [60].

For the present study, we fix the number of inputs and outputs of the nodes so that the light cones of all nodes have the same shape. We study the simplest nontrivial case, in which all nodes have exactly two inputs and two outputs. For these networks, the greatest possible sensitivity is $\lambda = 2$.

Shalizi's complexity measure, along with concepts of light cone and causal states, for a spatially extended system is previously introduced in Section 2.3. Ref. [17] outlined an algorithm for computationally distinguishing the causal states of cellular automata, and we can implement similar procedures to distinguish the causal states of our RBNs. Because nodes in RBNs are not assigned spatial positions, the definition of past and future light cones, L^- and L^+, requires some technical modifications. In the calculation of C_μ, $L_\mu^-(i,t)$ of node i at time step t is defined as

$$\begin{aligned} L_\mu^-(i,t) &= \{(X(j,u),\ t-u,\ d(i,j)) \\ &\qquad \forall\ u < t \text{ and } d(i,j) \leqslant t-u\}, \end{aligned} \tag{4.4}$$

where *d(i, j)* is the shortest distance between nodes i and j (the minimum number of links that must be traversed to get from node i to node j), and the speed of propagation of information is now 1. By this definition, different nodes that have the same distance from a given node are deemed indistinguishable for the purpose of identifying a light cone configuration. Although the update rules (Eq. 4.1) contain indices for node inputs, the calculation of statistical complexity is defined without reference to the underlying physical laws and hence without the knowledge of which input is which at any given node.

For calculating the complexity $C_\nu(i)$ of an individual node, however, different inputs and outputs are distinguishable. When comparing two past or future light cone configurations of the same node i at different time steps, one can keep track of which input is which and observe that when the two inputs differ, the future light cone configuration depends upon which input value is 1. Consequently, $L_\nu^-(i,t)$ is defined as

$$L_\nu^-(i,t) = \{(X(j,u),\ t-u,\ j)\ \forall\ u < t \text{ and } d(i,j) \leqslant t-u\}. \tag{4.5}$$

In measuring C_μ or C_ν for a RBN in the limit of large system size, as opposed to a regular lattice, it is sufficient to restrict the computation to light cones of depth one. The the lack of memory in the update rules ensures that all the relevant information for determining the state of a node is determined by the configuration one time step in the past, and the lack of short loops in the random graph ensures that all information propagating from a given past light cone to a given future light cone must pass through the single node under consideration. Fig. 4.1 illustrates the latter point. In a regular lattice, a node's past light cone can influence its future light cone through multiple paths, whereas in an RBN, the chance of finding such a path is vanishingly small in the limit of large system size. Thus, even though the definition of statistical complexity relies on arbitrarily deep light cones, the results for depth 1

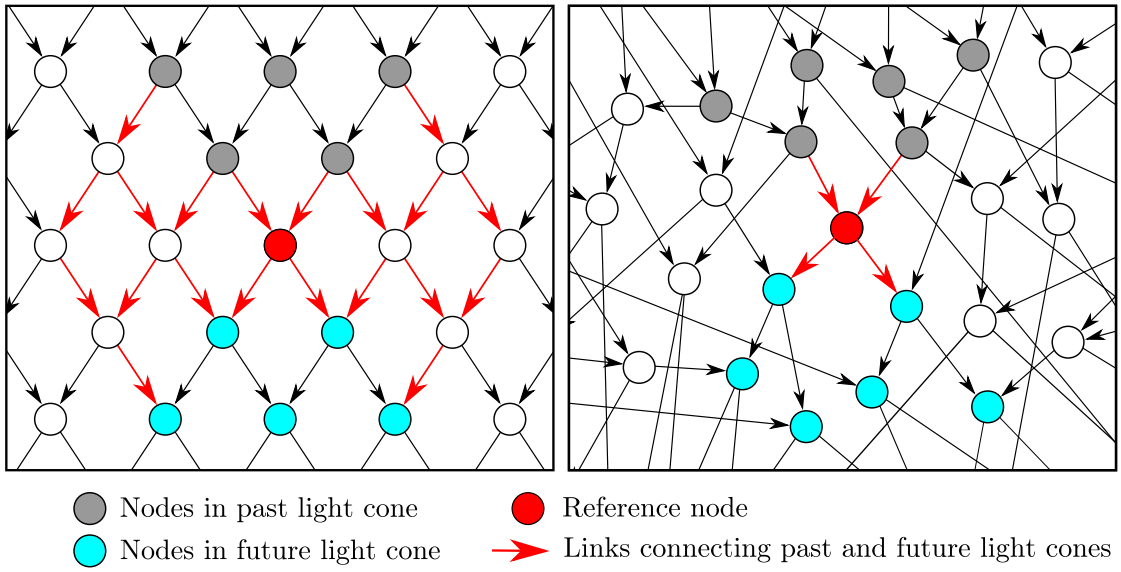

FIGURE 4.1: Local portions of (a) a two-dimensional regular lattice network and (b) a random network. Grey/green nodes are in the past/future light cones that are truncated at a depth of two time steps. The light cone configuration includes the value of each node at a time corresponding to the number of links that must be crossed to reach the reference node. In (a), a node's past light cone influences its future light cone through multiple paths. In (b) the past light cone influences the future light cone only through the reference node itself.

become exact for RBNs with system size $N \to \infty$. This fact enables us to calculate C_μ and, in some cases, C_ν analytically in the large system limit.

C_μ is calculated from depth-1 light cones as follows. For a node with two inputs i and j, there are only three possible past light cone configurations: $\{0,0\}$, $\{1,1\}$, and $\{1,0\}$ or $\{0,1\}$ (note we have omitted the relative time and distance entries in writing the light cone configurations because they are now trivial after we restrict the light cone depth to 1), occurring with probability $(1-\rho)^2$, ρ^2, and $2\rho(1-\rho)$, respectively, and yielding probabilities for the reference node being ON of $\langle f(0,0)\rangle$, $\langle f(1,1)\rangle$, and $1/2(\langle f(0,1)\rangle + \langle f(1,0)\rangle)$ respectively. As discussed above, the probability of observing a future light cone configuration, L^+, depends only on the state of the reference node. Thus, if each of the three possible L^-'s yields a unique probability

for the reference node to be ON or OFF, which in turn yields a unique distribution of L^+'s, then each L^- is itself a causal state and we have

$$\begin{aligned} C_\mu &= -\rho^2 \log_2(\rho^2) - (1-\rho)^2 \log_2((1-\rho)^2) \\ &\quad -2\rho(1-\rho)\log_2(2\rho(1-\rho))\,. \end{aligned} \tag{4.6}$$

Modifications to Eq. 4.6 are required if the above assumptions do not hold. For example, if any two of $\langle f(0,0)\rangle$, $1/2(\langle f(0,1)\rangle + \langle f(1,0)\rangle)$ and $\langle f(1,1)\rangle$ are equal, then there would be less than three causal states. The following scenario is also possible. If the state of a reference node x is 1, the probability that an output is 1 is

$$\rho_1^+ = \rho\langle f(1,1)\rangle + 1/2(1-\rho)\langle f(1,0)\rangle + 1/2(1-\rho)\langle f(0,1)\rangle\,; \tag{4.7}$$

and if $x = 0$, the probability that an output is 1 is

$$\rho_0^+ = (1-\rho)\langle f(0,0)\rangle + 1/2\rho\langle f(1,0)\rangle + 1/2\rho\langle f(0,1)\rangle\,. \tag{4.8}$$

In the case of an accidental degeneracy $\rho_1^+ = \rho_0^+$, the distribution of L^+ is independent of x and therefore independent of L^-, in which case all three possible L^- collapse to a single causal state, yielding $C_\mu = 0$.

Fig. 4.2 shows the comparison between the analytical calculation of C_μ and simulation results for ensembles of networks with mixtures of two logic functions. The horizontal axis denotes the fraction of nodes that get assigned the indicated Boolean function. The ensemble in Fig. 4.2(a) satisfies the requirements for Eq. 4.6. The ensemble in Fig. 4.2(b) only has two causal states because $\langle f(0,0)\rangle$=$\langle f(1,1)\rangle$=$1-q$, and $1/2(\langle f(0,1)\rangle + \langle f(1,0)\rangle)$=1, where q is the fraction of XOR nodes in a XOR-ON network. The first two terms in Eq. 4.6 collaps into one for such an ensemble, and the corresponding C_μ is given by

$$\begin{aligned} C_\mu &= -(\rho^2 + (1-\rho)^2)\log_2(\rho^2 + (1-\rho)^2) \\ &\quad -2\rho(1-\rho)\log_2(2\rho(1-\rho)). \end{aligned} \tag{4.9}$$

From Eq. 4.2 we can obtain the fixed-state bias for a XOR-ON network to be

$$\rho = \frac{-1 + 2q + \sqrt{1 + 4q - 4q^2}}{4q}, \tag{4.10}$$

which is monotonically decreasing from $\rho = 1$ to $\rho = 0.5$ for $q \in [0, 1]$. We could subsequently obtain a complicated expression of C_μ in terms of q which we would omit here, but we can see that $C_\mu(q)$ is an increasing function in $q \in [0, 1]$ because $C_\mu(\rho)$ monotonically decreases in $\rho \in [0.5, 1]$. Simulation results indicate that C_μ does not change noticeably for a given ensemble for network sizes above $N = 1000$. The simulations for $N = 10^4$ agree well with the large system analytical result. The disagreements at $q = 0.05$ and $q = 1$ in the Fig. 4.2b are due, respectively, to the fact that differences between causal states are too small for the simulations to resolve ($1-q$ too close to q) and to a collapse of the type described in the previous paragraph ($\rho_1^+ = \rho_0^+$). For some choices of Boolean functions, the bias ρ can oscillate, which leads to persistent oscillations in C_μ.

Critical networks (with $\lambda = 1$) have been hypothesized to have properties that might be favored by natural selection or other self-organized processes [24]. Our findings show that C_μ is not simply related to sensitivity, so that maximization of C_μ does not correspond to selection of critical networks. Fig. 4.2 illustrates this point with two examples. In Fig. 4.2(a), we see that the network is critical at both $q = 0$ and $q = 0.5$ and that C_μ is a maximum in one case but zero in the other. In Fig. 4.2(b), we see that C_μ increases monotonically as λ increases from 1 (the critical value) to 2.

The fact that C_μ can be high in ordered systems ($\lambda < 1$), where the attractor dynamics is trivial, highlights the fact that the spatial inhomogeneity of states alone can produce a high complexity. In these networks, almost all nodes are frozen on a fixed value, independent of the initial conditions, but different nodes may be frozen

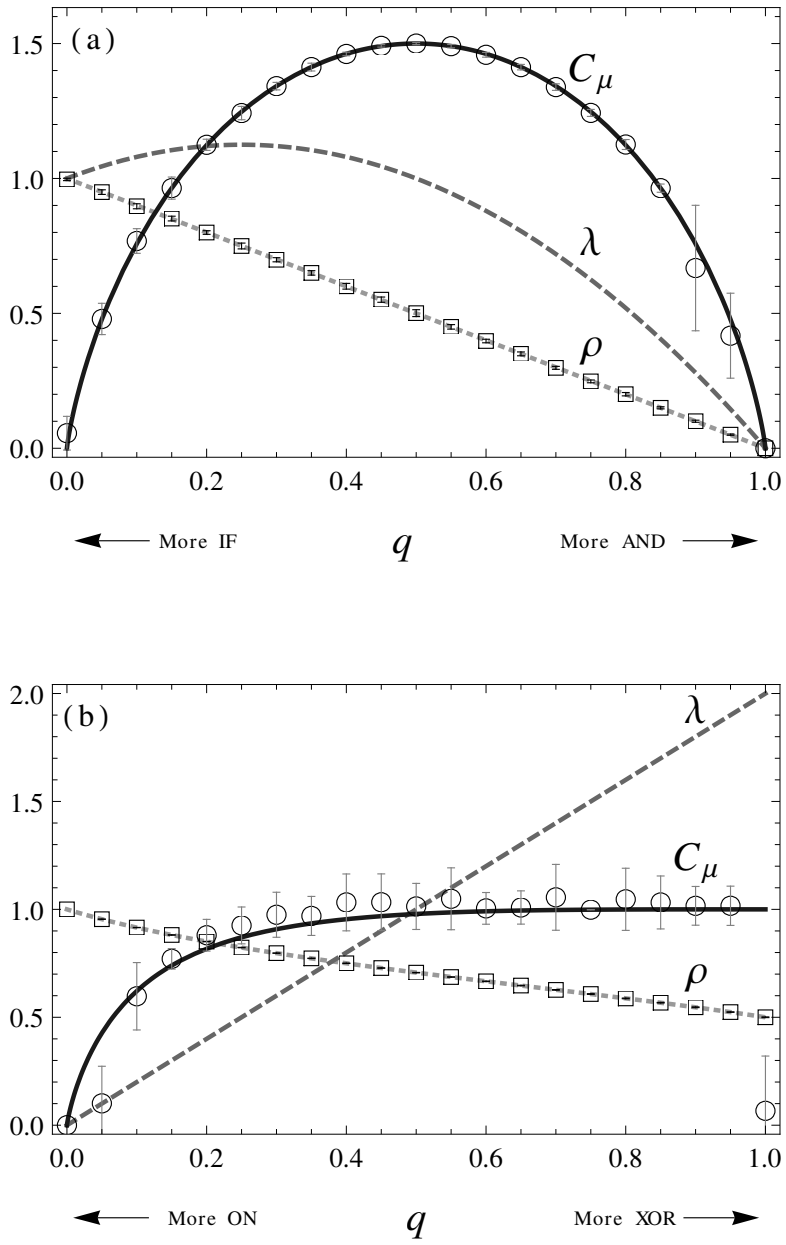

FIGURE 4.2: The steady-state bias ρ, sensitivity λ and complexity C_μ for networks consisting of two types of nodes. (a) A fraction q of the nodes are assigned the AND function $(f(0,0), f(0,1), f(1,0), f(1,1)) = (0,0,0,1)$ while the others are assigned $(1,0,1,1)$ (IF). (b) A fraction q are assigned $(0,1,1,0)$ (XOR) and the rest $(1,1,1,1)$ (always ON). The solid curve and circular points are respectively the analytical and simulation results for C_μ. The size of networks is $N = 10^4$, and time of data collection for each network realization is $T = 10^6$. For each of the 21 ratios of the two logic functions (21 dots in the graph), 30 RBNs are constructed. For each RBN, 30 runs are simulated with different initial conditions.

on different values and C_μ becomes a measure of the degree of variation from node to node. The fact that C_μ can be high in strongly disordered systems (λ near the maximum possible value of 2), reflects the tendency of the bias ρ to approach values that maximize C_μ at these points.

The calculation of C_ν treats each component as an agent with its own causal states and complexity, then averages those complexity values. When causal states are determined by considering the past and future light cones of a single node at different times, every frozen node has zero complexity, as does any node for which the past and future light cones are uncorrelated. In ordered or critical networks, where only a vanishingly small number of nodes are not frozen, C_ν is very close to zero. In highly disordered systems, the behavior of most nodes closely approximates a purely stochastic process, so C_ν is again near zero. Thus C_ν is maximized somewhere in the disordered regime.

Fig. 4.3 shows a typical plot of complexity C_ν as a function of sensitivity λ. Recall that $\lambda = 2$ is the highest sensitivity value possible for a system in which each node has exactly two outputs. All ensembles of systems with a full range of sensitivity values exhibit the same general relation between C_ν and λ, with C_ν maximized at different λ from ensemble to ensemble.

We have not found a way to calculate C_ν analytically for networks with a general combination of logic functions, but we can do it (in the large system limit) for the special case of the networks represented in Fig. 4.2(b), which contain only the logic functions XOR (0110) and ON (1111). In this case, the complexity of each node is either 0 or 1, depending on whether it and/or its neighbors are frozen or not.

The first step in calculating C_ν is to find the fraction γ of nodes that are frozen. In general, for large networks of two-input logic functions in which the probability that a node is frozen when a subset of its inputs are frozen is independent of the

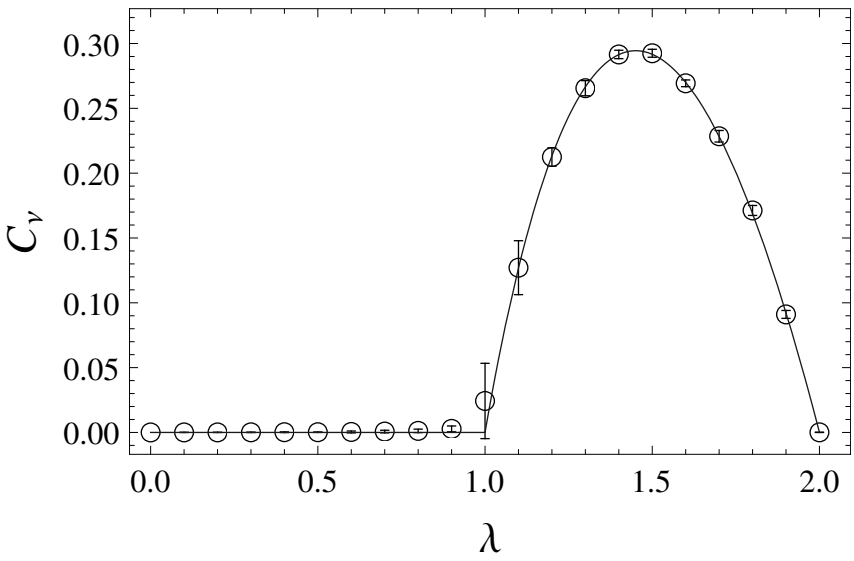

FIGURE 4.3: A typical pattern for complexity C_ν vs sensitivity λ. C_ν is close to zero for $\lambda \leqslant 1$ and for the maximum value of $\lambda = 2$. C_ν is maximized near $\lambda = 1.5$. The two logic functions are $(0, 1, 1, 0)$ and $(1, 1, 1, 1)$, as in Fig. 4.2(b). The solid curve shows theoretical results and the circular dots show simulation results. The size of networks is $N = 10^4$ and time of data collection for each network realization is $T = 10^6$. For each of the 21 ratios of the two logic functions (21 dots in the graph), 30 RBNs are constructed. For each RBN, 30 runs are simulated with different initial conditions.

value of those frozen inputs, γ is given by a solution of the equation

$$p_0(1-x)^2 + 2p_1x(1-x) + p_2x^2 = x\,, \tag{4.11}$$

where p_k is the probability that a node will be frozen if exactly k of its inputs are frozen. (See [61], Eq. (70).) The XOR-ON networks satisfy the requirements and have $p_0 = p_1 = 1 - q$ and $p_2 = 1$, where q is the fraction of XOR nodes, which yields

$$\gamma = (1-q)/q \text{ or } 1\,. \tag{4.12}$$

For $q > 1/2$, $\gamma = (1-q)/q$ is the stable solution. For $q < 1/2$, $\gamma = 1$ is the stable solution, implying that the fraction of frozen nodes approaches unity.

The unfrozen nodes are all XORs, some of which have one frozen input and therefore act either as a copier or inverter of the other input. The network of unfrozen

nodes thus acts as a network in which every node executes a parity function (or its inversion). We propose the following general conjecture about such networks.

Conjecture: *Every node in a synchronously updated Boolean network containing only parity functions (for arbitrary numbers of inputs to each node) will have bias of exactly* $1/2$ *when averaged over trajectories originating from all possible initial states of the network, where the bias of an individual node at a given time* t *is defined as the fraction of trajectories for which that a node is ON at time* t*. Moreover, the bias is not affected if the nodes executing parity functions are embedded in a larger network in which all other nodes freeze after some transient.*

The conjecture is supported by numerical simulations of 10,000 randomly generated 16-node networks with only XOR and ON gates. We find that the bias of exactly 1/2 on each unfrozen node is maintained on each individual time step. For many of these networks, it can be shown that every state of the unfrozen portion has exactly one pre-image, which is enough to guarantee that all states occur with equal frequency when averaged over initial conditions. In cases where some states have two pre-images, the number of recurrent states decreases by a factor of two (and may in some cases be reduced by additional factors of 2).

Our conjecture provides the basis for a computation of C_ν. First note that in the large system limit, all frozen nodes have complexity $C_\nu(i) = 0$. For an unfrozen node in the XOR-ON network, $C_\nu(i)$ may be 0, which occurs when an unfrozen XOR has outputs only to unfrozen nodes that all have additional unfrozen inputs. The value of each output node in this case is equally likely to take either value, no matter what the value of node i. $C_\nu(i)$ may be nonzero, however, if the unfrozen node has at least one output to a node whose other input is frozen. The future light cone distribution then depends on x_i, and the past light cones that yield the two values occur with equal probability, which gives $C_\nu(i) = 1$.

Calculating C_ν for the XOR-ON ensemble of networks comes down to determining

the fraction of nodes that satisfy the condition for having $C_\nu(i) = 1$, which can be obtained from a mean-field calculation as follows. Let X_1 and X_2 be inputs of node X_3 and assume that X_1 has no other output. If we assume that X_1 is an unfrozen node, then the probability that X_2 is frozen and X_3 is unfrozen is $q\gamma$. Because the only way for X_1 to have complexity 1 is for X_2 to be frozen, the probability that X_1 has complexity 1 is $q\gamma$. For an unfrozen node with two outputs, the complexity will be 1 if either of the outputs has a frozen input, which occurs with probability $1 - (1 - q\gamma)^2$. Thus the C_ν is equal to the total fraction of nodes with nonzero complexity:

$$C_\nu = (1 - \gamma)(1 - (1 - q\gamma)^2)\,. \tag{4.13}$$

Using Eq. 4.12, we have the system average

$$C_\nu = \frac{1}{q}(1 - q^2)(2q - 1)\,. \tag{4.14}$$

Eq. 4.14 agrees well with simulation results (Fig. 4.3).

The XOR-ON example shows that C_ν can be calculated for specific distributions of logic functions and, more importantly, illustrates that the complexity of an individual component depends on globally determined dynamics, not just on the logical process carried out by the individual node or on the smaller network motifs containing the node. The pattern of frozen nodes is generated by a transient process that may propagate through the entire network [61].

The precise import of the value of $C_\nu(i)$ at a given node is not immediately clear. We have measured the correlation between $C_\nu(i)$ and a new measure $\delta(i)$ that characterizes the effect of replacing node i with a random number generator. $\delta(i)$ is determined as follows. Two realizations of the same network are run in parallel with the same initial condition. After running long enough for transients to decay, the logic function f_i is ignored in one of the copies and node i is replaced by a stochastic

agent generating $x_i = 1$ or 0 with probabilities p and $1 - p$, respectively, on each time step. Over the course of M steps, we calculate the fraction of time that each node in the network is ON for each of the realizations, forming two N-dimensional vectors. The Euclidean distance between these two vectors is denoted $\delta(i, p)$, and we define $\delta(i) \equiv \min\{\delta(i, p) : p \in [0, 1]\}$. We obtain an approximate measure of $\delta(i)$ by taking $M = 10N$ and considering $p = 0.1\,n$ for $n = 0, 1, \ldots, 10$. A study of the XOR-AND and IF-AND ensembles shows that the correlation coefficient between $C_\nu(i)$ and $\delta(i)$ in the disordered regime ranges between 0.4 to 0.7 for network size N=1000. This suggests an interpretation of $C_\nu(i)$: it is a measure of importance of the logical processing performed at node i for determining the global dynamics. A low value of $C_\nu(i)$ means that the computations done by node i can be effectively simulated by a random number generator.

We note that correlations associated with the existence of two paths in the network from one node to another can cause frozen nodes to have nonzero $C_\nu(i)$. The correlation arises when a path of unfrozen nodes from some node j passes through the frozen node i and then immediately to an unfrozen node k, while another path of unfrozen nodes of the same length goes from j to k without passing through i. Let j_1 be the input to i along the first path. Then x_{j1} and x_k may be correlated due to the common source at node j, in spite of the fact that node i does not pass on any information. (See Fig. 4.4.) In the large system limit, the fraction of nodes affected by this type of correlation vanishes.

Because $\delta(i)$ is necessarily zero for any frozen node, the existence of multiple paths in a finite system causes some nodes with high $C_\nu(i)$ to have low $\delta(i)$. We further note that $\delta(i)$ itself is not a perfect measure of dynamical importance of a given node's activity because the average Euclidean distance between the bias vectors may be small even though the sequence of states is substantially different. In fact, we observe that $\delta(i)$ saturates at low $C_\nu(i)$ values. Comparing $C_\nu(i)$ with more precise

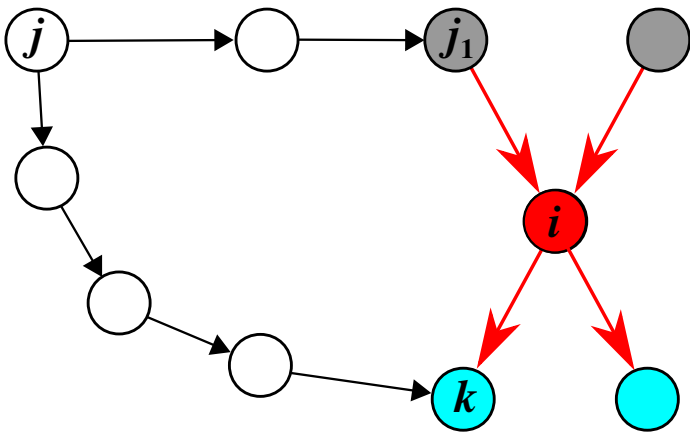

FIGURE 4.4: An example of a network structure that can cause a frozen nodes to have nonzero $C_\nu(i)$: node i is frozen, but its past and future light cones are correlated because nodes j_1 and k receive input from the same node j through chains of unfrozen nodes that differ in length by exactly two links.

measures of dynamical importance would be an interesting topic for future research.

4.2 Conclusion

In extending the formalism of Shalizi's complexity measure to random Boolean networks, a distinction must be made between determining causal states by averaging over nodes at a given time step versus averaging over time for each node separately. The two methods are equivalent for systems described by the same input-output function at every node of a uniform lattice, but yield different results for systems that are spatially inhomogeneous either because the input-output functions are different for different nodes or because the network topology is not a regular lattice. The networks we have studied have both types of inhomogeneity.

We find that C_μ, obtained from statistics at a single time step, can be calculated analytically for RBN's, and that it is not simply related to sensitivity to small perturbations. The maximum of C_μ can occur in the ordered, disordered, or critical regime, depending on the details of the probability distribution chosen for the Boolean rules

in the network. C_μ is directly related to the steady state (or long-term oscillatory) bias in the node values.

C_ν, obtained from statistics compiled over time for individual nodes, is zero (in the large system limit) everywhere in the ordered and at the limiting value of sensitivity in the disordered regime, so it is maximized somewhere in the disordered regime. The value at an individual node, $C_\nu(i)$, may be interpreted as a measure of the role that node in determining the global dynamics. Nodes that can be replaced by random number generators without substantially altering the global dynamics (and hence perform no essential information processing) tend to have lower $C_\nu(i)$. This last finding may be helpful in characterizing the behavior of social or biological systems and the individual agents or components that comprise them. Interestingly, the $C_\nu(i)$'s depend on the global features of the network that determine its attractors, not just on the local logic functions and topology. In other words, the identification of important players in a network requires global information about the network, not just a characterization of each individual's behavior.

5

Semi-annealed Approximation in A Social Network Model with Short Loops

In modeling real systems with a large number of interacting agents using a Boolean network model, a quantity of common interest is the bias - the average fraction of nodes in the ON state. The bias is particularly important in social networks, when often the main task of the research effort is to predict population statistics such as the fraction of people that are going to vote for a certain candidate, adopt a certain ideology, or get infected with a contagious disease. The bias calculation for large random Boolean networks (RBNs) using the annealed approximation technique was previously discussed in Sections 2.1.2 and 4.1. The topologies of social networks are very different from the random network topologies we have considered in Section 2.1 and Chapter 4, and the difference may render the analytical method for calculating the bias developed earlier less applicable when studying models of social networks.

One feature that is commonly present in social networks that previously discussed random network models are lacking is *reciprocity* [62, 63, 64]: if person A influences the opinion or behavior of person B through social interactions, then person B would

likely be conversely influenced by person A. Reciprocity is represented by two nodes with outputs directed at each other, or in other words, a loop of length 2. Loops that are slightly larger (*e.g.* of lengths 3 and 4) are common in social networks as well. In particular, the presence of triangular structures (i.e. loops of length 3) has been well studied using a measure called clustering coefficient that equals the ratio of the number of triangles to the number of connected triplets [65, 66, 67]. It is reported in these studies that the clustering coefficient in social networks such as friendship networks and co-authorship networks is higher than in random networks with the same average degree. It has also been widely demonstrated that many social networks have topologies characterized by loosely connected "communities," where each community is a group of nodes amongst which the level of connectivity is high and short loops are abundant [68, 69, 70, 67].

Additionally, loops of length 1, or self inputs, are a generic feature for a broad family of models, including networks with time-reversible computation [71] and gene regulatory networks with self-regulation [72]. In the synchronous updating scheme, a self-input indicates that a node's state at time t depends on its own state at the last time step $t-1$. Such a memory effect is often present in social dynamics, where an individual's behavior often does not only depend on that of its peers, but also on its own state as well as negotiations between its earlier behavior and that of its peers.

The probability of finding one of these short loops is vanishingly small in a random network model where every node's inputs are uniformly drawn from the whole network, as we previously stated in Chapter 4. The bias calculation for RBNs using the annealed approximation introduced earlier assumes that all edges and logic functions are constantly reassigned, effectively ignoring any state correlation. However, state correlation is a feature that can strongly affect the bias of nodes in short loops. To see why, consider for example a node i with two inputs i_1 and i_2 that themselves

share a common input. Now states of i_1 and i_2 are correlated because of the common input, and thus when calculating the probability that node i is ON, we can no longer assume statistical independence amongst its inputs and do the probability multiplications as we do so in the bias map given by Eq. 2.6 [73]. Given that short loops are found in a broad class of networks, it is desirable to devise a new technique to obtain the network bias that accommodates the presence of short loops.

In this chapter, we introduce a semi-annealed approximation technique to calculate the bias of a class of synchronous Boolean networks in which every link is in a short loop. This approximation takes into account the state correlation between any two nodes that are connected by a link. The networks of interest are a class of social game-inspired networks that feature short loops of lengths 1 and 2 on every node. Individual nodes' update rules are dependent on their expectations of the outcome from the prisoner's dilemma game they play with their neighbors. We demonstrate that our semi-annealed approximation produces bias predictions that are significantly closer to the simulated data than the traditional annealed approximation technique does.

Some results of game theoretic interest are found as a byproduct of this study. An individual's average reward could change with the change of its own expectation and the expectation distribution in the network. How the changes are correlated may not be obvious. Some nontrivial collective behaviors have been observed and are reported at the end of the chapter. The semi-annealed approximation accounts for most of the reported results.

This chapter presents original work of the author, performed under the guidance of James Moody and Joshua E. S. Socolar.

player 1 \ player 2	Defect	Cooperate
Defect	A	C
Cooperate	B	D

Table 5.1: Reward matrix for player 1 in a single game.

5.1 Model description – a social game-inspired network model

We consider a model where prisoner's dilemma game (PDG) is played between all pairs of connected nodes in a K-regular random network. The prisoners' dilemma game is a two-player game in which each player can choose to "defect" against the other player or "cooperate" with the other player. Each player gets a reward based on what he and his opponent did. The reward values, given in Tab. 5.1, satsify $C > D > A > B$, yielding an interesting property that no matter what the other player does, it is always to the benefit of a player to defect, but if both players rationally choose to defect, they would each receive a lower reward than if they both have chosen to cooperate. Many studies, including the current one, also require $2D > B + C$, implying that the maximum shared reward happens when both players cooperate. A K-regular random network is a network drawn uniformly from the set of all undirected networks where each node has exactly K neighbors. The network topology is quenched and the game is played at discrete time steps. At any given time step, each node plays the same strategy, either defecting (OFF) or cooperating (ON), against all its neighbors, and receives a reward in each of the K interactions as given by the reward matrix of the PDG. The dynamics of each node is determined by a quenched expectation threshold θ_i: if at time t node i does not receive a total reward that exceeds the expectation θ_i, it would play the opposite strategy at time $t+1$ than at t. This update scheme is essentially a Boolean network evolution where each node has a logic function that takes arguments from its neighbors and itself.

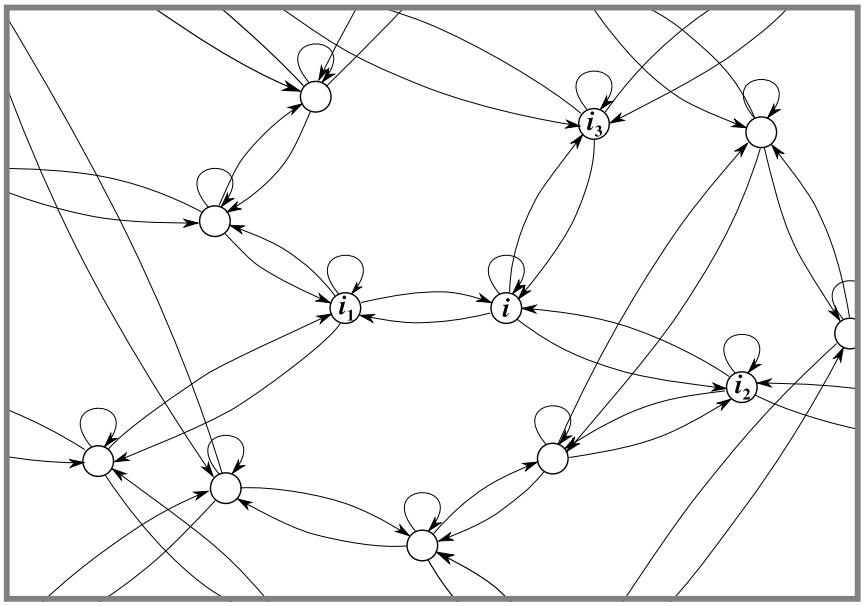

FIGURE 5.1: An illustration of local features of a prisoners' dilemma game (PDG) network where each node has $K = 3$ neighbors.

In the directed network picture used for other networks in this thesis, every node in the current PDG network model would have K input links and K output links connecting to the same K nodes, because players come in pairs to play one another and influences one another's strategy in the next time step; every node also has a self-input, representing the fact that a player's strategy depends on how its previous strategy plays out. Each inter-nodal link is in a loop of length 2 and each node has a self-input that is a loop of length 1. The network is filled with short loops, examples of the reciprocity and memory effects found in a large class of real networks. An illustration of local features of the $K = 3$ PDG network is given in Fig. 5.1.

We also note that cellular automata, a vastly popular mathematical model, in its original and most prevalent form incorporate both reciprocity and a memory effect [74, 75, 76]. Cellular automata have been used in the modeling of snow flake formation, crystallization, and a wide class of ecological, industrial and artificial intelligence problems (see [77] for a review). Our model is different from cellular

automata in that the network topology is random instead of being a lattice, which brings it one step closer to characterizing real networks. Thus, even though the current model is motivated by a particular social game, its applicability may be wider.

Taking the self-inputs into account, we can express the update rule for the PDG network as

$$x_i(t) = f_i(x_i(t-1), x_{i_1}(t-1), x_{i_2}(t-1), \ldots, x_{i_K}(t-1)), \tag{5.1}$$

where i_1, i_2, ..., i_K are neighbors of node i, and the effective number of inputs per node is $K+1$. Because our model dictates that each node plays the same strategy against all its neighbors at a given time step and the reward matrix is the same for all interactions, the Boolean update rule would treat each neighbor equally, and only the sum of one's neighbors' states matters, not their sequence. We can thus rewrite Eq. 5.1 as

$$x_i(t) = f_i^* \left(x_i(t-1), \sum_{j=1}^{K} x_{i_j}(t-1) \right) . \tag{5.2}$$

The particular function f_i of node i depends on its expectation threshold θ_i. We note that the number of possible Boolean functions for a node reduces drastically from the model in Eq. 5.1 to the model in Eq. 5.2 for $K \geqslant 3$. In the former model, the number of possible Boolean functions is the same as RBNs with $K+1$ inputs: $2^{2^{(K+1)}}$, whereas in the latter model the number if $2^{2+(K+1)} = 2^{3+K}$. The number of possible Boolean functions could be further restricted by the rule that a node i only reverses its strategy if its reward is below its expectation threshold. The reward for i after a round of playing is determined by x_i, $\sum_{j=1}^{K} x_{i_j}$, and the universal reward matrix. Using the general reward matrix of a single game given by Tab. 5.2, we can

write out the reward for a round of playing for i as

$$\mathfrak{R}_i\left(x_i, \sum_{j=1}^{K} x_{i_j}\right) = r_{x_i 1} \sum_{j=1}^{K} x_{i_j} + r_{x_i 0}(K - \sum_{j=1}^{K} x_{i_j}) = r_{x_i 0} + (r_{x_i 1} - r_{x_i 0}) \sum_{j=1}^{K} x_{i_j}, \quad (5.3)$$

and we can also write out the Boolean function f_i more explicitly:

$$f_i^*\left(x_i, \sum_{j=1}^{K} x_{i_j}\right) = x_i \oplus \Theta\left[\theta_i - \mathfrak{R}_i\right] = x_i \oplus \Theta\left[\theta_i - r_{x_i 0} - (r_{x_i 1} - r_{x_i 0}) \sum_{j=1}^{K} x_{i_j}\right]. \quad (5.4)$$

Here the Heaviside function Θ yields 1 if and only if the expectation is not met. The XOR function, $\oplus$,indicates a switch of strategy in that case.

We now show a more concrete example of the model. In this network, $K = 3$ and the PDG has the single-play reward matrix $r_{00} = r_{10} = 0$, $r_{11} = 1$ and $r_{10} = R, (1 < R < 2)$. This simplified reward matrix has been shown to preserve all the interesting features of game [78]. The possible rewards a node can have at any time step are given in Tab. 5.3. Also included in in Tab. 5.3 are 3 possible Boolean functions and their corresponding expectation thresholds θ_i. The Heaviside function in Eq. 5.4 implies that any θ_i value is equivalent to other θ_i values in its neighborhood in defining the Boolean function, barring some boundary cases. The three ranges of θ_i that define distinct Boolean function in Tab. 5.3 demonstrate this point. From the reward column, we can see that $0, 1, 2, 3, R, 2R$, and $3R$ are the only possible rewards a player can get after each round. If $R < 1.5$, the sorted sequence of these numbers is $0, 1, R, 2, 2R, 3, 3R$, and if $R > 1.5$, the sorted sequence is $0, 1, R, 2, 3, 2R, 3R$. In either case, there are 6 intervals between 0 and $3R$ that correspond to distinct Boolean functions. We will focus on the former case where $1 < R < 1.5$. We will ignore the negative expectation case $\theta_i < 0$ when a node's state never changes, and we will also ignore the case with unrealistic expectation $\theta > 3R$ where a player is never content and constantly switching its strategy. Individuals in a network can

x_i \ x_{i_j}	0	1
0	r_{00}	r_{01}
1	r_{10}	r_{11}

Table 5.2: Reward matrix for player i from a single play. State 0 represents defection and state 1 represents cooperation.

states at t			x_i(t+1)			
x_i	$x_{i_1} + x_{i_2} + x_{i_3}$	reward	if $\theta_i \in (0,1)$	if $\theta_i \in (1,R)$	if $\theta_i \in (R,2)$	...
0	0	0	1	1	1	
0	1	R	0	0	1	
0	2	$2R$	0	0	0	
0	3	$3R$	0	0	0	
1	0	0	0	0	0	
1	1	1	1	0	0	
1	2	2	1	1	1	
1	3	3	1	1	1	

Table 5.3: Rewards from a round of playing and Boolean function table for a network with $K = 3$ and reward matrix $r_{00} = r_{10} = 0$, $r_{11} = 1$ and $r_{10} = R, (1 < R < 2)$.

have the same expectation threshold or different thresholds. We will consider with different threshold distributions in the next section.

There are many interesting game theoretic questions one can ask about this model. For example, in the case where every player has the same expectation threshold, what is the threshold value that maximizes the total network reward, and if a distribution of threshold values is allowed in the same network, which threshold values would best rewarded, and how does that change with the distribution? We explore some of these problems in Section 5.3, but we will first focus on developing an analytical tool that calculates the network bias and the average reward.

5.2 Annealed and semi-annealed approximations on the PDG network

The annealed approximation, when applied to RBNs, has been shown to predict the bias of a network very accurately. We recall that the bias of a RBN is the fixed point of the bias map given in Eq. 2.6:

$$\rho(t+1) = \left\langle \sum_{\mathbf{x}} f_i(\mathbf{x}) \rho(t)^{|\mathbf{x}|} (1-\rho(t))^{K-|\mathbf{x}|} \right\rangle_i ,$$

where $\mathbf{x} = \{x_i, x_{i_1}, x_{i_2}, \ldots, x_{i_K}\}$ is a vector of possible input states. The model described in the Section 5.1 is very different from RBNs. In a RBN, each node's quenched inputs are randomly chosen, and there is no correlation between the indices of a node's inputs and outputs. In the current model, however, each link is paired with a link between the same nodes in the opposite direction, so a node's inputs and outputs are the same set of nodes. Each node also has a self-input. Because of these topological differences, we need to be careful about how to apply the old analytical techniques in the new model. The annealed approximation applied to a RBN assumes that in each time step all links are rewired, effectively assigning each node a new set of uncorrelated inputs and outputs. Rewiring in the new model should stay true to some key model assumptions: we should not decouple the input and output links, and we should not rewire the self-inputs. The annealed approximation would instead choose a new set of K neighbors for each node at every time step, and connect each node to its neighbors with bi-directional links.

We will for now first concentrate on networks with uniform expectation thresholds (and thus uniform logic functions). For this type of network, our caution in the rewiring process would not actually make a difference when it comes to calculating the bias. In a network where all nodes have the same connectivity rules and logic functions, we assume that they all have the same bias, which is also the network

bias. Let this bias at time t be $\rho(t)$. The probability that a node's input sequence being $\mathbf{x} = x_i, x_{i_1}, x_{i_2}, \ldots, x_{i_K}$ is then

$$\Pr(\mathbf{x}|\rho) = \rho(t)^{|\mathbf{x}|}(1-\rho(t))^{K+1-|\mathbf{x}|}\,, \tag{5.5}$$

a very similar expression to Eq. 2.4 used in the RBN case, embedded in Eq. 2.6 shown above. The only difference is made by the fact that now the number of inputs is $K+1$. The bias map generated by the annealed approximation thus treats RBNs and the uniform-logic PDG networks equally. We can rewrite Eq. 2.6 in terms of $x = x_i$, $y = \sum_{j=1}^{K} x_{i_j}$, and the logic functions f^* given in Eq. 5.4:

$$\rho(t+1) = \left\langle \sum_{x,y} f_i^*(x,y)\rho(t)^{x+y}(1-\rho(t))^{K+1-x-y} \right\rangle_i . \tag{5.6}$$

Note that we can drop the angle bracket and the subscript i because we are considering uniform logic networks. Even though Eq. 5.6 and Eq. 2.6 are bias maps that employ the same approximation technique, their performances from application to the respective classes of networks are quite different. We have seen in Fig. 4.1 that the annealed approximation predicts the network bias of RBNs very accurately. In the current case, the same analytical technique often produces bias values that have significant error margins from the simulated data. The network example from Section 5.1 has 6 threshold intervals that corresponds to distinct Boolean functions. We simulate uniform-logic networks with each one of these 6 functions to obtain their bias, and we also calculate their bias using the annealed approximation. The results are shown in Tab. 5.4. The annealed approximation yields smaller bias than simulations in every case. The average deviation from the simulated data is -17.6%, and the largest deviation is -40.5%. The root-mean-square deviation is 0.156.

It is not surprising that the annealed approximation does relatively poorly at predicting the network bias in our PDG network as compared to in RBNs. The

	steady-state bias ρ^* in uniform-logic networks					
θ_i interval	$(0,1)$	$(1,R)$	$(R,2)$	$(2,2R)$	$(2R,3)$	$(3,3R)$
simulation	0.5718	0.4362	0.8402	0.4843	0.5024	0.5
annealed approximation	0.5	0.3028	0.5	0.4064	0.5	0.4722
semi-annealed approximation	0.6696	0.3793	0.9828	0.4793	0.5182	0.5361

Table 5.4: Steady-state bias for the network example given in Section 5.1. Bias values are obtained from simulation, annealed approximation, and semi-annealed approximation. Simulations are done on 30 networks of size 10,000, each initialized 30 times with different initial conditions and run over 100,000 time steps.

approximation scheme assumes no state correlation between different inputs of a node, and this is a valid assumption in large RBNs with vanishing fraction of nodes in short loops. In our PDG network, the abundance of short loops gives rise to state correlations and thus renders the annealed approximation less applicable. To make this point clearer, consider the local network structures shown in Fig. 5.2. The structure in Fig. 5.2(a) may be taken from a RBN where almost always local structures are tree-like, and inputs of a node i do not share inputs of their own. This is one feature that allows the application of annealed approximation, which assumes the statistical independence of all inputs of a node: $P(x_{i_1}, x_{i_1}) = P(x_{i_1})P(x_{i_2})$. This multiplying of probabilities is used in the derivation of the bias map shown in Eq. 5.6. However, when the network structure deviates from locally tree-like to having short loops, such as in Figs. 5.2(b), 5.2(c), and 5.2(d), the assumption of statistical independence of a node's input no longer holds, because now a node's input may share some common inputs of their own. Specifically in the current PDG network model where a node i has inputs $i, i_1, i_2, \ldots, i_K$, because of the reciprocity and self-inputs in the model, these nodes share a common input node, i, so their states are correlated. The correlation becomes stronger when the number of inputs

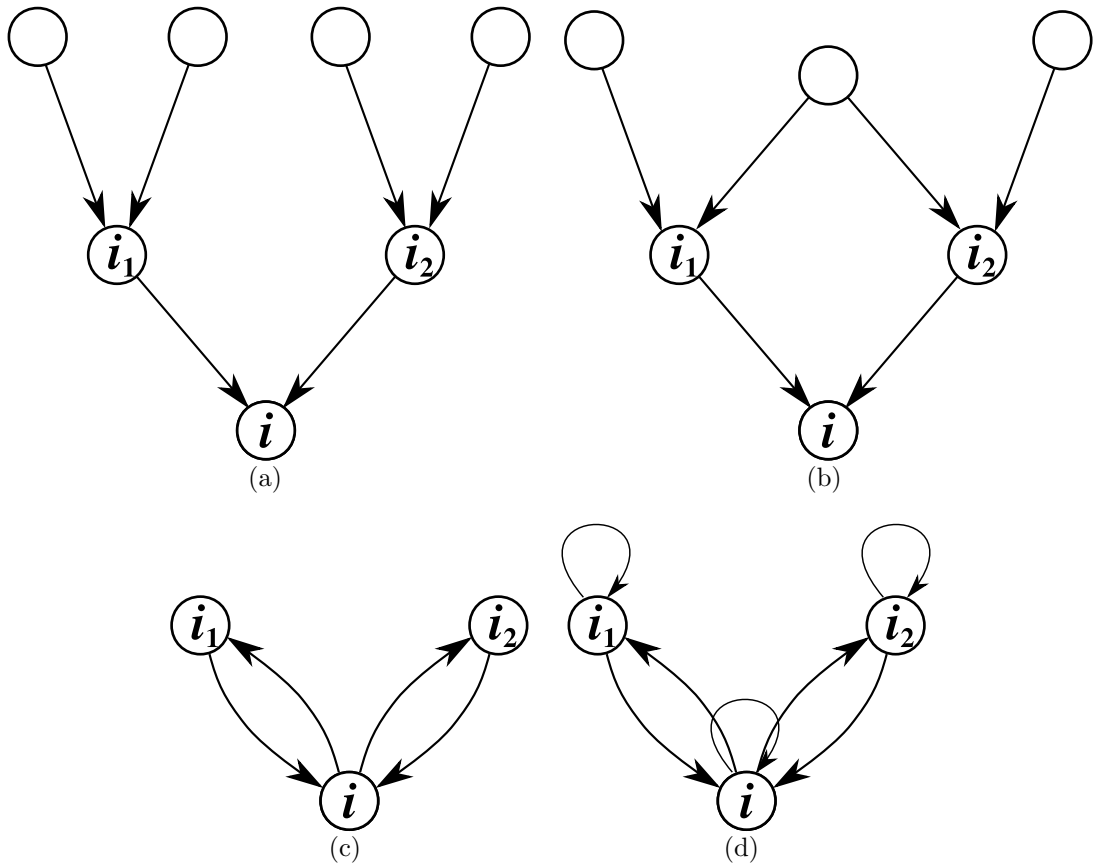

FIGURE 5.2: (a) an RBN, where local structures are tree-like; (b) a feed-forward look that ends at node i; (c) reciprocity: each link is coupled with another link in the opposite direction; (d) same as the PDG network where reciprocity and memory effect are incorporated.

per node gets smaller. The stronger the correlation, the less valid it is to assume statistical independence between the nodes' states and to multiply probabilities as we do in Eq. 5.5. Now we know that the state of i is correlated with the state of each neighbor i_j, but i_j's state is further correlated with those of its own other $K-1$ neighbors that are not connected to i, so it is likely that i's state is correlated with these $K-1$ nodes', and subsequently with their other neighbors' states as well, even though we expect the correlation to be weaker and weaker with longer distance. To improve upon the bias map generated by the annealed approximation, we can take

some short-range correlations into consideration. The first step is to consider the correlation between a node and its neighbors. We now introduce an analytical tool that does exactly that.

Consider again a randomly chosen node and its neighbors: $i, i_1, i_2, \ldots, i_K$. We would subsequently call this collection the i-neighborhood, and call the collection of nodes in i-neighborhood sans i the i-neighbors. From Section 5.1, particularly Eq. 5.2 and Eq. 5.4, we know that a node's neighbors are treated equally when it comes to determining the node's state. So instead of representing the state of a neighborhood by the state sequence $\mathbf{x} = x_i, x_{i_1}, x_{i_2}, \ldots, x_{i_K}$, we introduce a new notation for a neighborhood state $s^i = S_{xy}$, where s^i is the literal for i-neighborhood state, and S_{xy} indicates the specific state configuration where $x = x_i$ and $y = \sum_{j=1}^{K} x_{i_j}$. Additionally, We denote p^i_{xy} the probability that s^i assumes the value S_{xy}. Note x can take the values 0 or 1, and y can take the values $0, 1, \ldots, K$, so there are $2(K+1)$ p^i_{xy}'s, which sum to 1. Let them form a vector $\mathbf{p}^i = (p^i_{00}, p^i_{01}, \ldots, p^i_{0K}, p^i_{10}, \ldots, p^i_{1K})^T$. There is a vector $\mathbf{p}^i$ associated with every neighborhood, just as there is a bias value ρ_i for every node. The bias map from annealed approximation assumes that every node has the same bias ρ. We will now instead assume that every neighborhood has the same probability vector $\mathbf{p}$. This technique takes into account the intra-neighborhood correlations and ignores any longer-range correlation. It is therefore a *semi-annealed* approximation. We note that pairwise and neighborhood analyses have been used before in stochastic network models, *e.g.* in [79, 80, 15]. This work takes the neighborhood approach to a class of deterministic networks. The goal is to come up with a map $G(\mathbf{p})$ such that

$$\mathbf{p}(t+1) = G(\mathbf{p}(t)). \tag{5.7}$$

The steady-state probabilities $\mathbf{p}^* = (p^*_{00}, p^*_{01}, \ldots, p^*_{0K}, p^*_{10}, \ldots, p^*_{1K})^T$ would be found

by solving for $\mathbf{p}^*$ in

$$\mathbf{p}^* = G(\mathbf{p}^*). \tag{5.8}$$

The steady-state bias can then be obtained by a simple summation:

$$\rho^* = \sum_{y=0}^{K} p^*_{1y}. \tag{5.9}$$

The challenge now is to find an expression for the map $G(\mathbf{p})$, and we will achieve this goal in careful steps. We first note that Eq. 5.7 is actually a list of $2(K+1)$ equations, each of which can be written in the form $p_{xy}(t+1) = G_{xy}(\mathbf{p}(t))$. The problem then becomes finding a general expression for $G_{xy}(\mathbf{p})$.

Given $\mathbf{p}^i(t)$, the distribution of the i-neighborhood states $s^i(t) = S_{xy}$, we can obtain the distribution of $x_i(t+1)$, because for every S_{xy}, $x_i(t+1)$ is known for certain: $x_i(t+1) = f_i^*(x, y)$. The uncertainty in the new neighborhood state $s^i(t+1)$ thus lies in the neighbors, $y_i(t+1)$. To obtain $y_i(t+1)$, we would need to know the neighbors' neighborhood states at t. The semi-annealed approximation assumes that these neighborhoods' share the same state distribution as the i-neighborhood. However, any two neighbors' neighborhood states are correlated because they share some of the same nodes. For example, in Fig. 5.2 where $K = 3$, the i and j-neighborhoods share nodes i and j. Even if we assume $\mathbf{p}^j(t) = \mathbf{p}^i(t)$, we need to take into account that for every specific $s^i(t) = S_{xy}$, the distribution of $s^j(t)$ is restricted. In other words, $\Pr(s^j(t) = S_{ab} | s^i(t) = S_{xy})$ is different from $\mathbf{p}^i(t) = \Pr(s^j(t))$.

If we have already obtained an expression for the conditional adjacent neighborhood state distribution $\Pr(s^j(t) = S_{ab} | s^i(t) = S_{xy})$ (to be derived soon), then we can use it to calculate the probability that a neighboring node of i is ON at the next time step. Simplifying the notation, we let $P^j_{0\to 1}$ denote the probability that a neighbor j

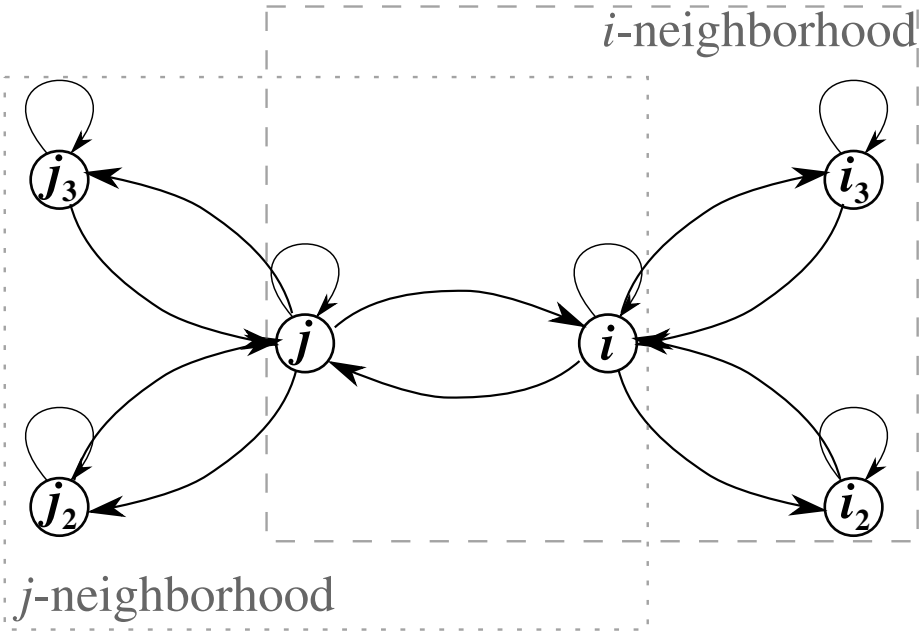

FIGURE 5.3: Two adjacent neighborhoods in a $K = 3$ network. When taking the semi-annealed assumption that the two neighborhoods share the same state distribution, caution should be taken to account for the fact that the two neighborhoods have overlapping nodes and thus have correlated states

is ON at time $t+1$ had it been OFF time at t:

$$\begin{aligned} P^j_{0\to 1} &\equiv \Pr\left(x_j(t+1)=1 \mid x_j(t)=0, s^i(t)=S_{xy}\right) \\ &= \frac{\Pr\left(x_j=0, x_j(t+1)=0 | s^i(t)=S_{xy}\right)}{\Pr(x_j(t)=0)} \\ &= \frac{\sum_b \Pr\left(s^j(t)=S_{0b} | s^i(t)=S_{xy}\right) f^*(0,b)}{\sum_b \Pr\left(s^j(t)=S_{0b} | s^i(t)=S_{xy}\right)}, \end{aligned} \tag{5.10}$$

where f^* is the uniform logic function of the network. Networks with mixed logic functions will be dealt with later. Similarly, we obtain the probability that a neighbor

j continues to be ON at time $t+1$ if it was already ON at t to be

$$P^j_{1\to 1} \equiv \Pr\left(x_j(t+1)=1 \mid x_j(t)=1, s^i(t)=S_{xy}\right) = \frac{\sum_b \Pr\left(s^j(t)=S_{1b}|s^i(t)=S_{xy}\right) f^*(1,b)}{\sum_b \Pr\left(s^j(t)=S_{1b}|s^i(t)=S_{xy}\right)}. \tag{5.11}$$

For $s^i(t)=S_{xy}$, we know y of the K nodes in i-neighbors are ON and $K-y$ of them are OFF. If the number of ON nodes in the i-neighbors is $y_i(t+1)=y'$ at the next time step, we know some of these y' nodes may have been switched from being OFF and others would have been ON. Using combinatorics, we can find the distribution for $y_i(t+1)$:

$$\Pr\left(y_i(t+1)=y'|s^i(t)=S_{xy}\right) = \sum_{y''=0}^{y'} \binom{y}{y''} (P^j_{1\to 1})^{y''}(1-P^j_{1\to 1})^{y-y''} \binom{k-y}{y'-y''} (P^j_{0\to 1})^{y'-y''}(1-P^j_{0\to 1})^{(K-y)-(y'-y'')}, \tag{5.12}$$

where y'' in each addend stands for the exact number of ON neighbors that continue to stay ON. Now because we already know $x_i(t+1)=f^*_i(x,y)$, we can now arrive at the distribution of i-neighborhood state at $t+1$:

$$\Pr\left(s^i(t+1)=S_{x'y'}|s^i(t)=S_{xy}\right) = (x' \odot f^*(x,y))\Pr\left(y_i(t+1)=y' \mid s^i(t)=S_{xy}\right), \tag{5.13}$$

where $\odot$ is the XNOR operation that yields 1 when the two operands are the same and 0 otherwise. Equation 5.13 shows the transitional probabilities between neighborhood states. Summing the transition probabilities from all neighborhood states to unity, we can then express the probability distribution of i-neighborhood state at

$t+1$ as

$$\begin{aligned} p^i_{xy}(t+1) &= \Pr(s^i(t+1) = S_{xy}) \\ &= \sum_{x'y'} \Pr\left(s^i(t+1) = S_{xy} \mid s^i(t) = S_{x'y'}\right) \Pr(s^i(t) = S_{x'y'}) \\ &= \sum_{x'y'} \Pr\left(s^i(t+1) = S_{xy} \mid s^i(t) = S_{x'y'}\right) p^i_{x'y'}(t)\,. \end{aligned} \tag{5.14}$$

Equation 5.14 represents $2(K+1)$ equations (one for each p_{xy} in vector $\mathbf{p}$) that make up an iterated map of the neighborhood state distributions. It is the explicit form of Eq. 5.7 that we set out to obtain.

We have omitted one important step in the derivation. That is the expression for conditional adjacent neighborhood state distribution $\Pr(s^j(t) = S_{ab}|s^i(t) = S_{xy})$. The derivation for a general expression of this probability is cumbersome. We first show a calculation with a specific example to demonstrate the reasoning used in the derivation. Consider Fig. 5.2 again where $K = 3$. If the i-neighborhood is in state S_{11}, we want to know the probability of the j-neighborhood being in state S_{02}: $\Pr(s^j(t) = S_{02}|s^i(t) = S_{11})$. Writing this expression more explicitly, and using Bayes' theorem, we obtain the following:

$$\begin{aligned} &\Pr\left(s^j(t) = S_{02} \mid s^i(t) = S_{11}\right) \\ =&\Pr\left(x_j(t) = 0, y_j(t) = 2 \mid x_i(t) = 1, y_i(t) = 1\right) \\ =&\Pr\left(x_j(t) = 0 \mid x_i(t) = 1, y_i(t) = 1\right) \Pr\left(y_j(t) = 2 \mid x_j(t) = 0, x_i(t) = 1, y_i(t) = 1\right)\,. \end{aligned} \tag{5.15}$$

Because $y_i(t) = 1$ means that 2 out of the 3 nodes in i-neighbors are OFF, and because j is one of those nodes, we can deduce

$$\Pr(x_j(t) = 0 \mid x_i(t) = 1, y_i(t) = 1) = \Pr(x_j(t) = 0 \mid y_i(t) = 1) = \frac{2}{3}\,. \tag{5.16}$$

That takes care of the first probability in the last line of Eq. 5.15. The second probability is more tricky. We recognize that because x_i is 1 and i is one of the

nodes in the j-neighbors, we then know $y_j(t)$ is at least 1, and thus possible values of $y_j(t)$ are limited to 1, 2, and 3. It is tempting to consider this limitation to be the only condition that $\{x_j(t) = 0, x_i(t) = 1, y_i(t) = 1\}$ imposes on $y_j(t)$ and draw an equivalence between $\Pr(y_j(t) = 2 \mid x_j(t) = 0, x_i(t) = 1, y_i(t) = 1)$ and the following equation:

$$\begin{aligned}\Pr(y_j(t) = 2 \mid x_j(t) = 0, y_j(t) \geqslant 1) &= \frac{\Pr(y_j(t) = 2, x_j(t) = 0, y_j(t) \geqslant 1)}{\Pr(x_j(t) = 0, y_j(t) \geqslant 1)} \\ &= \frac{p_{02}}{p_{01} + p_{02} + p_{03}}\,. \end{aligned} \tag{5.17}$$

However, while Eq. 5.17 is a correct equation, it does not provide the right conditional probability for $y_j(t) = 2$ in our case. Note that $y_j(t) \geqslant 1$ is equivalent to saying at least one of j's neighbor is ON, but our case says that a particular one of j's neighbors, i, is ON. There is a difference between the two statements even though we treat all the neighbors equally. Treating all neighbors equally dictates that the probability that a particular neighbor i of j is ON is equal to the probability that a *randomly selected* neighbor of j is ON. This probability, compared to the one for at least one neighbor being ON, is biased towards the neighborhood states with more ON nodes among the neighbors. For example, it is twice as likely to randomly select a ON neighbor in state S_{02} as it is in S_{01}. If we let j_k be a randomly selected neighbor of j, then we have

$$\begin{aligned}\Pr(y_j(t) = 2 \mid x_j(t) = 0, x_i(t) = 1, y_i(t) = 1) &= \Pr(y_j(t) = 2 \mid x_j(t) = 0, x_{j_k}(t) = 1) \\ &= \frac{2p_{02}}{p_{01} + 2p_{02} + 3p_{03}}\,. \end{aligned} \tag{5.18}$$

Summarizing equations 5.15, 5.16 and 5.18, we arrive at the conclusion of this example:

$$\Pr(s^j(t) = S_{02} | s^i(t) = S_{11}) = \frac{4p_{02}}{3p_{01} + 6p_{02} + 9p_{03}}\,. \tag{5.19}$$

Applying the same reasoning used in the above example to other neighborhood states and K values, we can obtain a general expression for the conditional adjacent neighborhood state distribution:

$$\begin{aligned}
&\Pr(s^j(t) = S_{ab} \mid s^i(t) = S_{xy}) \\
&= \left((1 \odot a)\frac{y}{K} + (0 \odot a)\frac{K-y}{K} \right) \frac{((1 \odot x)b + (0 \odot x)(K-b))\, p_{ab}}{\sum_{b'=0}^{K} ((1 \odot x)b' + (0 \odot x)(K-b'))\, p_{ab'}} .
\end{aligned} \tag{5.20}$$

Equation 5.20 fills in the last piece in our derivation of the iterative map for neighborhood state distribution $\mathbf{p}$ using semi-annealed approximation (Eq. 5.7). We show three explicit expressions of a such a map in Fig. 5.4, where the equations pertain to two cases given in Tab. 5.3. The network bias is obtained with Eq. 5.9. Analytical solutions to Eq. 5.7 are usually difficult to obtain, so they are often found numerically. The bias values from semi-annealed approximation for $K = 3$ networks with uniform logic are shown in Tab. 5.4, along with the results from the annealed approximation. We can see that going from the annealed approximation to the semi-annealed approximation, the mean percentage deviation is 5.07%, and the mean-square deviation is 0.0761, respectively less than a third and a half of the deviations generated by the annealed approximation. We also note that while the annealed approximation predicts lower than simulated bias for each of the 6 rules, the semi-annealed approximation have errors more evenly distributed around 0.

From a social network modeling perspective, we are also interested in the reward that is associated with each expectation threshold. The reward of a single round of play for an individual with a known neighborhood state is given in Eq. 5.3. Averaging the rewards of different nodes' neighbor states, we can obtain the average reward for the whole network. The steady-state neighborhood state distribution from the semi-annealed approximation is given by the vector $\mathbf{p}$. From the annealed approximation, we know the steady-state bias ρ, and we can use it to obtain the expected

$$
\begin{pmatrix} p00 \\ p01 \\ p02 \\ p03 \\ p10 \\ p11 \\ p12 \\ p13 \end{pmatrix} = \begin{pmatrix}
0 & \frac{3\,(2\,p01+p02)^2\,p10}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{9\,(2\,p01+p02)\,p10^2}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{27\,p10^3}{(3\,p10+2\,p11+p12)^3} & 1 & 0 & 0 & 0 \\
0 & \frac{(2\,p01+p02)\,(18\,p00\,p10+(2\,p01+p02)\,(2\,p11+p12))}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{3\,p10\,(9\,p00\,p10+2\,(2\,p01+p02)\,(2\,p11+p12))}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{27\,p10^2\,(2\,p11+p12)}{(3\,p10+2\,p11+p12)^3} & 0 & 0 & 0 & 0 \\
0 & \frac{3\,p00\,(9\,p00\,p10+2\,(2\,p01+p02)\,(2\,p11+p12))}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{(2\,p11+p12)\,(18\,p00\,p10+(2\,p01+p02)\,(2\,p11+p12))}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{9\,p10\,(2\,p11+p12)^2}{(3\,p10+2\,p11+p12)^3} & 0 & 0 & 0 & 0 \\
0 & \frac{9\,p00^2\,(2\,p11+p12)}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{3\,p00\,(2\,p11+p12)^2}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{(2\,p11+p12)^3}{(3\,p10+2\,p11+p12)^3} & 0 & 0 & 0 & 0 \\
\frac{(2\,p01+p02)^3}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{9\,p00\,(2\,p01+p02)^2}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\frac{27\,p00^2\,(2\,p01+p02)}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\frac{27\,p00^3}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix} \begin{pmatrix} p00 \\ p01 \\ p02 \\ p03 \\ p10 \\ p11 \\ p12 \\ p13 \end{pmatrix}
$$

$$
\begin{pmatrix} p00 \\ p01 \\ p02 \\ p03 \\ p10 \\ p11 \\ p12 \\ p13 \end{pmatrix} = \begin{pmatrix}
0 & \frac{(2\,p01+p02)^2\,(3\,p10+2\,p11)}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{(2\,p01+p02)\,(3\,p10+2\,p11)^2}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{(3\,p10+2\,p11)^3}{(3\,p10+2\,p11+p12)^3} & 1 & \frac{p11}{p11+2\,p12+3\,p13} & 0 & 0 \\
0 & \frac{(2\,p01+p02)\,(6\,p00\,(3\,p10+2\,p11)+p12\,(2\,p01+p02))}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{(3\,p10+2\,p11)\,(9\,p00\,p10+6\,p00\,p11+4\,p01\,p12+2\,p02\,p12)}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{3\,p12\,(3\,p10+2\,p11)^2}{(3\,p10+2\,p11+p12)^3} & 0 & \frac{2\,p12+3\,p13}{p11+2\,p12+3\,p13} & 0 & 0 \\
0 & \frac{3\,p00\,(9\,p00\,p10+6\,p00\,p11+4\,p01\,p12+2\,p02\,p12)}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{p12\,(6\,p00\,(3\,p10+2\,p11)+p12\,(2\,p01+p02))}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{3\,p12^2\,(3\,p10+2\,p11)}{(3\,p10+2\,p11+p12)^3} & 0 & 0 & 0 & 0 \\
0 & \frac{9\,p00^2\,p12}{(3\,p00+2\,p01+p02)^2\,(3\,p10+2\,p11+p12)} & \frac{3\,p00\,p12^2}{(3\,p00+2\,p01+p02)\,(3\,p10+2\,p11+p12)^2} & \frac{p12^3}{(3\,p10+2\,p11+p12)^3} & 0 & 0 & 0 & 0 \\
\frac{(2\,p01+p02)^3}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 0 & \frac{p11^2}{(p11+2\,p12+3\,p13)^2} & \frac{p11^3}{(p11+2\,p12+3\,p13)^3} \\
\frac{9\,p00\,(2\,p01+p02)^2}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 0 & \frac{p11\,(4\,p12+6\,p13)}{(p11+2\,p12+3\,p13)^2} & \frac{3\,p11^2\,(2\,p12+3\,p13)}{(p11+2\,p12+3\,p13)^3} \\
\frac{27\,p00^2\,(2\,p01+p02)}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 0 & \frac{(2\,p12+3\,p13)^2}{(p11+2\,p12+3\,p13)^2} & \frac{3\,p11\,(2\,p12+3\,p13)^2}{(p11+2\,p12+3\,p13)^3} \\
\frac{27\,p00^3}{(3\,p00+2\,p01+p02)^3} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{(2\,p12+3\,p13)^3}{(p11+2\,p12+3\,p13)^3}
\end{pmatrix} \begin{pmatrix} p00 \\ p01 \\ p02 \\ p03 \\ p10 \\ p11 \\ p12 \\ p13 \end{pmatrix}
$$

FIGURE 5.4: Two examples of iterative maps of neighborhood state distribution $\mathbf{p}$. These equations are explicit forms of Eq. 5.7. They pertain to cases covered in Tab. 5.3 where $K = 3$ and all nodes have the same logic functions given by (a). $\theta \in (0, 1)$; (b) $\theta \in (1, R)$;

neighborhood state distribution, again assuming no correlation between nodes in a neighborhood:

$$P(x, y|\rho) = (1 \odot x)\rho^{x+y}(1-\rho)^{K-y} + (1 \oplus x)\rho^{y}(1-\rho)^{1+K-y} \tag{5.21}$$

We can then calculate the expected reward per node for both types of approximation schemes:

$$\bar{\mathfrak{R}}_{\text{annealed}} = \sum_{x=0}^{1}\sum_{y=0}^{K} \mathfrak{R}(x, y)P(x, y|\rho)\,, \tag{5.22a}$$

$$\bar{\mathfrak{R}}_{\text{semi-annealed}} = \sum_{x=0}^{1}\sum_{y=0}^{K} \mathfrak{R}(x, y)p_{xy}\,. \tag{5.22b}$$

The average rewards per node for the network examples we have been using ($K = 3$, uniform expectation thresholds) are shown in Tab. 5.5. The root-mean-square deviations from the simulated results are approximately $0.408\sqrt{2.22 - 1.12R + 0.249R^2}$ and $0.408\sqrt{0.664 - 0.515r + 0.163R^2}$ for the annealed and semi-annealed approximations respectively, or in the ranges of (0.429, 0.474) and (0.207, 0.228) respectively for $R \in (1, 1.5)$. The latter is less than half of the former for each R. Figure 5.5 shows rewards plotted against expectation thresholds for a specific R value ($R = 1.25$). Some qualitative observations from these plots also hold true with other R values: the semi-annealed approximation produces the right ordering of expectation ranges in terms of highest to lowest reward and predicts correctly whether reward exceeds expectation in every range, while the annealed approximation fails at both tasks.

We have focused on networks with uniform logic functions. Now we consider the more general case where a distribution of logic functions are allowed in a network. We cautioned earlier that the rewiring process in applying the annealed approximation on our PDG network should leave the self inputs intact. While this caution does not change the bias map for networks with uniform logic, because every node is assumed

Average reward $\bar{\mathfrak{R}}$ in uniform-logic networks

θ_i interval	$(0,1)$	$(1,R)$	$(R,2)$	$(2,2R)$	$(2R,3)$	$(3,3R)$
simulation	$0.897 + 0.818R$	$0.652+0.657R$	$2.13+0.389R$	$0.909+0.544R$	$0.659+0.848R$	$1.02+0.482R$
annealed approximation	$0.75+0.75R$	$0.275+0.633R$	$0.75+0.75R$	$1.04+0.727R$	$0.75+0.75R$	$0.669+0.748R$
semi-annealed approximation	$1.04+0.965R$	$0.553+0.585R$	$2.89+0.0537R$	$0.886+0.552R$	$0.860+0.694R$	$1.13+0.483R$

Table 5.5: Steady-state average reward per node per time step for the network example given in Section 5.1. Rewards are obtained from simulation, annealed approximation, and semi-annealed approximation. Simulations are done on 30 networks of size 10,000, each initialized 30 times with different initial conditions and run over 100,000 time steps. Model assumptions define the range of R to be $(1, 1.5)$.

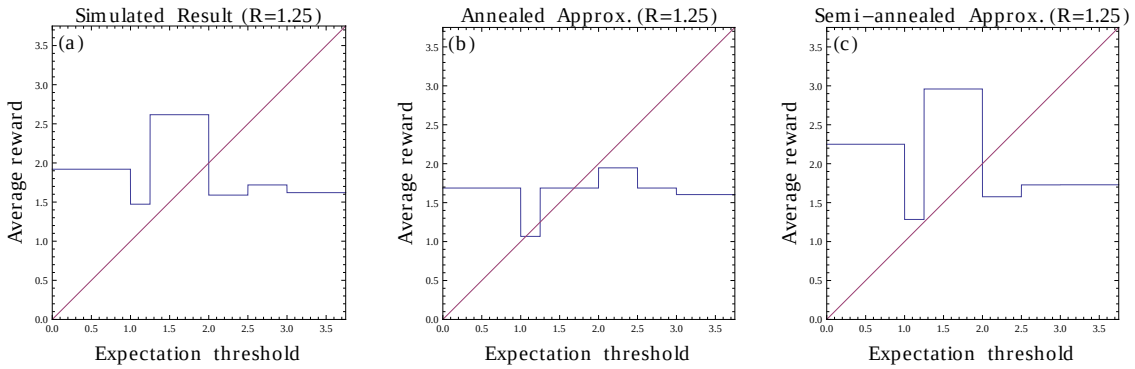

FIGURE 5.5: Average reward per node as a function of expectation threshold for our example of $K = 3$ uniform-logic networks (detailed network description given in Section 5.1). The reward parameter R is chosen to be 1.25 for these graphs. It can be observed that the semi-annealed approximation produces the right ordering of expectation range in terms of highest to lowest reward and predicts correctly whether reward exceeds expectation in every expectation range, while the annealed approximation fails at both tasks.

to have the same bias, it does make a difference in mixed logic networks. Every node's self input plays a unique role in its own logic function, compared with its neighbors that are considered equivalent, as can be seen in Eq. 5.4. Therefore, when nodes in a network have different logic functions, the self-input can reinforce the differences in the average bias of nodes with different logic functions, and it is desirable to assign a unique bias to each logic function. We can still treat each neighbor's bias as the network average. Detailed steps are shown below.

Let there be M types of logic functions $F_1, F_2, \ldots, F_M$ in a network taking a fraction of $\phi_1, \phi_2, \ldots, \phi_M$ of the nodes respectively. Each F_l is in the form of f^* in Eq. 5.4. Assign each F_l a bias π_l, and let the network average bias be ρ. We then have the following relations:

$$\pi_l(t+1) = \sum_{x,y} F_l(x,y)\pi_l(t)^x(1-\pi_i(t))^{1-x}\rho(t)^y(1-\rho(t))^{K-y}, \tag{5.23a}$$

$$\rho(t) = \sum_{l=1}^{M} \pi_l(t)\phi_l \,. \tag{5.23b}$$

Equation 5.23a represents a series of M bias maps, and when combined with Eq. 5.23b, gives us $M+1$ equations with $M+1$ variables. The steady-state solutions to these equations are the network bias and a bias for each type of logic functions.

Applying the semi-annealed approximation to networks with mixed rules follows a similar strategy. We assign each type of logic function F_l a neighborhood state probability vector $\mathbf{\Pi}^l = \pi^l_{00}, \pi^l_{01}, \ldots, \pi^l_{0K}, \pi^l_{10}, \ldots, \pi^l_{1K}$, and let the network average neighborhood state vector still be $\mathbf{p}$. For each node i with logic function F_l, we assume its neighborhood state distribution is $\mathbf{\Pi}^l$, and its neighbors' neighborhood state distribution is $\mathbf{p}$. Our semi-annealed treatment of uniform logic networks invoked the uniform logic function in Eqs. 5.10, 5.11 and 5.13, and these equations should now be modified to deal with mixed logic functions. Equation 5.10 can be rewritten as

$$\begin{aligned} P^j_{0\to 1} &= \frac{\sum_b \Pr\left(s^j(t) = S_{0b} | s^i(t) = S_{xy}\right) \left\langle f^*_j(0,b) \right\rangle_j}{\sum_b \Pr\left(s^j(t) = S_{0b} | s^i(t) = S_{xy}\right)} \\ &= \frac{\sum_l \sum_b \Pr\left(s^j(t) = S_{0b} | s^i(t) = S_{xy}\right) \phi_l F_l(0,b)}{\sum_b \Pr\left(s^j(t) = S_{0b} | s^i(t) = S_{xy}\right)} \,. \end{aligned} \tag{5.24}$$

Note that the equality between the first and second lines in Eq. 5.24 relies on the fact that $\Pr\left(s^j(t) = S_{0b} | s^i(t) = S_{xy}\right)$ is independent of the neighbor node index j. This is true because one's neighbors' neighborhood state distribution is still assumed to be $\mathbf{p}$. In other words, Eq. 5.20 is still valid. The equality between the second and third lines in Eq. 5.24 assumes that different logic functions are spread out evenly in the network, which is in agreement with the random network assumption. If, for example, nodes with the same logic functions tend to connect to each other, then the

averaging of f_j^* would need to be done differently for each logic function. Equations 5.11 and 5.13 are revised for mixed logic networks in similar fashion:

$$P_{1\to 1}^{j} = \frac{\sum_{l}\sum_{b} \Pr\left(s^j(t) = S_{1b} | s^i(t) = S_{xy}\right) \phi_l F_l(0,b)}{\sum_{b} \Pr\left(s^j(t) = S_{1b} | s^i(t) = S_{xy}\right)}\,, \tag{5.25}$$

$$\Pr\left(s^i(t+1) = S_{x'y'} | s^i(t) = S_{xy}\right) = \sum_{l} \phi_l(x' \odot F_l(x,y)) \Pr\left(y_i(t+1) = y' \mid s^i(t) = S_{xy}\right)\,. \tag{5.26}$$

We can now rewrite the neighborhood state distribution map Eq. 5.14 in terms of $\mathbf{\Pi}^l$ and $\mathbf{p}$:

$$\pi_{xy}^{l}(t+1) = \sum_{x'y'} \Pr\left(s^i(t+1) = S_{xy} \mid s^i(t) = S_{x'y'}\right) p_{x'y'}\,, \tag{5.27a}$$

$$p_{xy}(t) = \sum_{l} \phi_l \pi_{xy}^{l}(t)\,. \tag{5.27b}$$

Equations 5.23 and 5.27 are the bias maps and neighborhood state distribution maps for our PDG networks with mixed expectation thresholds. We note that we originally adopted the second strategy, the semi-annealed treatment because all of each node's neighbors' share an input and consequently state correlations within the neighbors arise and render the annealed approximation less applicable. However, when a node's neighbors have increasingly diverse logic functions, the correlation that arise from their shared input would become less pronounced, and thus the annealed approximation becomes more valid and the advantage of the semi-annealed approximation becomes less noticeable. For example, in the "most mixed" case of the $K = 3$ PDG network where the 6 logic functions make up a fraction of 1/6 each, the annealed and semi-annealed approximations both produce more accurate predictions than they do on uniform logic networks, and they perform comparably

well (see Fig. 5.6 for the case of $R = 1.25$), producing mean-root-square deviations of 0.0506 and 0.0454 in their bias predictions respectively. The semi-annealed treatment is, however, the only treatment that produces the correct ranking of expectation thresholds in terms of cooperativeness (fraction of players who cooperates, same as ρ). In the prediction of average reward, the two treatments again perform comparably, both yielding deviations from the simulated data around 0.15 ± 0.2 for all allowed values of R, with the annealed approximation actually edging out the semi-annealed approximation slightly for some R values (*e.g.* the $R = 1.25$ case shown in Fig. 5.7).

In a mixed logic network, logic functions do not have to be evenly distributed, and correlations can exist between a node's logic and those of its neighbors. This corresponds in the PDG network setting to the scenario where players with similar expectations tend to play each other, and more generally in other social networks this correlation can represent similarities amongst individuals in closely connected communities. We conjecture that the semi-annealed approximation would again be more useful in this scenario because correlations in logic among a node's neighbors leads to correlations in their states. Finding numerical evidence for this conjecture is left for future work.

5.3 Interesting game theoretic findings

Even though our study has centered on developing a semi-annealed approximation scheme to account for reciprocity and self-inputs in the random regular network topology, our examples of $K = 3$ prisoners' dilemma game networks have produced some interesting results of game theoretic interest. Results discussed below come from examinations of simulation outputs and can be reproduced by the semi-annealed approximation unless otherwise stated.

First of all, one may expect the cooperativeness to be high on uniform logic networks in which all players have low expectations just above 0. The state of

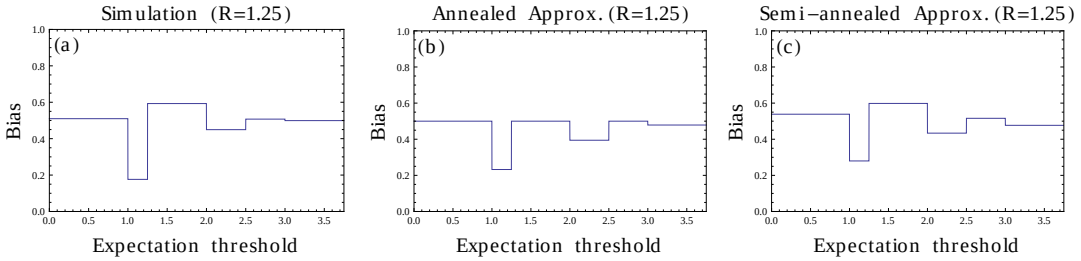

FIGURE 5.6: Average bias per node as a function of expectation threshold for $K = 3$ maximally mixed-logic networks, in which each of the 6 expectation thresholds makes up 1/6 of the players' strategies. The reward parameter R is chosen to be 1.25 for these graphs. The two approximation schemes perform comparably in terms of average deviation from the simulated data, but only the semi-annealed approximation predicts the correct order.

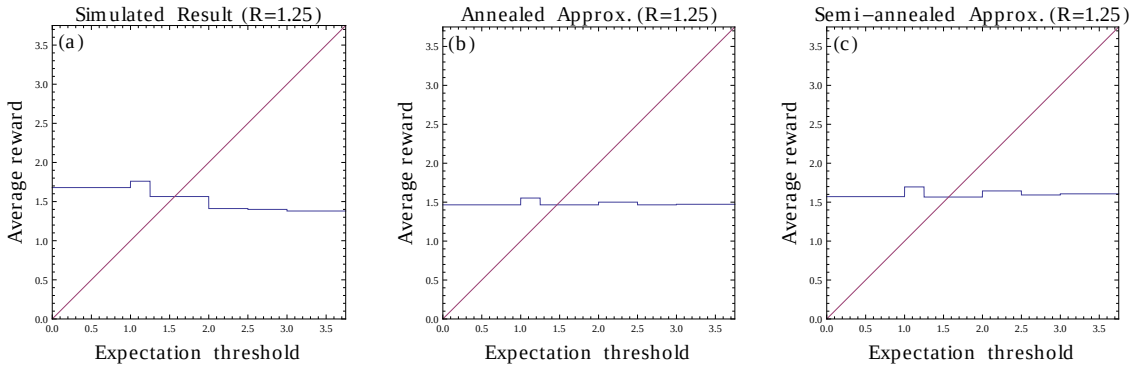

FIGURE 5.7: Average reward per node as a function of expectation threshold for $K = 3$ maximally mixed-logic networks, in which each of the 6 expectation thresholds makes up 1/6 of the players' strategies. The reward parameter R is chosen to be 1.25 for these graphs. The two approximation schemes perform comparably in terms of average deviation from the simulated data, and neither scheme predicts the correct order.

every player cooperating is a fixed-point of such a network. However, if a critical small fraction of players defect in the initial state, some cooperating players would become dissatisfied with their reward and start switching their strategies, and the fraction of cooperating players subsequently settles to a lower value (see Tab. 5.4 and Fig. 5.6(a)). For uniform logic network with expectation thresholds on the high end, say $R > 3$, players may be rarely satisfied and thus switch between cooperating and defecting frequently, yielding a bias of 0.5 for each node and for the whole network. The highest fraction of cooperating players in a uniform logic network is found in a medium expectation threshold range $\theta \in (R, 2)$. Intuitive reasoning is not easily obtained for this observation.

The ranking of the 6 rules in terms of cooperativeness does not change going from uniform logic networks to networks with 6 logic rules evenly distributed (refer to Tab. 5.4 and Fig. 5.6(a)). The ranking of their rewards, however, changes drastically (*e.g.* by comparing Figs. 5.5(a) and 5.7(a). Note the semi-approximation produces small deviations in both cases but does not produce the correct ordering in the latter case). The rule made by expectations of $\theta \in (1, R)$ is the least rewarded rule when used in uniform logic networks, but becomes the best rewarded in a mixed logic network. In both cases, it is the least cooperative. It is not surprising that players who are prone to defect are not well rewarded when playing amongst each other but do well for themselves when playing against players who are more prone to cooperate. It is interesting, though, to note that a player's expectation in this social game model is an indicator for its tendency to defect in both uniform and mixed logic networks. It is also not immediately clear why the particular expectation range of $\theta \in (1, R)$ is the least cooperative among all expectations when expectations immediately above or below are the two most cooperative ones. On the other hand, the best rewarded rule $\theta \in (R, 2)$ in the uniform logic case is the most cooperative rule.

In the maximally mixed logic network, players with expectation thresholds below

R on average have their expectation met while the ones with expectations above 2 are mostly dissatisfied. Moreover, the ones expecting higher rewards actually end up being less rewarded than their fellow players with lower expectations. Thus, if a satisfied player with relatively low expectation suddenly decides to raise its expectation, it could end up with lower rewards than the expectation it used to have. We have implemented such a single-player's change of heart in simulation and observed this exact effect with a number of logic distributions.

5.4 Conclusion

This chapter introduces a semi-annealed approximation scheme suitable for K-regular networks in which logic functions treat each input is equally. This approximation scheme takes into account the correlation of states on neighboring nodes, and generates accurate bias predictions for networks with reciprocity and memory effect, a task that traditional mean field calculations often fail. Taking a prisoners' dilemma game network model as the example in the study, the semi-annealed approximation, which keeps track of the neighborhood state distribution, provides a natural way of calculating the average reward from a round of game played between neighbors, and it helps explain some nontrivial observations of game theoretic interest.

More distributions of network connectivity and logic functions should be studied with the semi-annealed treatment in the future, especially cases where logic functions between neighboring nodes are correlated. This type of networks is more relevant to social studies, and it is more appropriate to evaluate with the semi-annealed approximation. Another measure of interest in random network studies, sensitivity to small perturbations, could also be readily obtained by considering the state transitions of a neighborhood, because now the question reduces to how many nodes in a neighborhood changes state at the next time step if the state of the center node is flipped. The detailed evaluation of sensitivity is left for future work as well.

6

Closing Remarks

The rise of network science over the last two decades is predicated on finding scientific insights by modeling systems of elements connected to one another through complex patterns of dynamical interactions. Because the network description is virtually content independent, the range of this field is exceedingly broad and incorporates many areas as diverse as physics, mathematics, biology, computer science, sociology and economics, just to name a few. Computational simulations and analytical evaluations of different models can help explain the origin of emergent properties observed in real systems, or provide a base line of null-hypothesis behavior in cases where more illuminating insights are not readily obtained. The successes described in the introduction were just a few among a rapidly increasing number of accomplishments achieved by network models. Network dynamics can be divided into two types, dynamics of the network structure and dynamics of the elements' states. Of the latter type, the Boolean abstraction is a popular strategy. In this thesis, we have focused on the study of structurally quenched Boolean networks, and the original research presented here contributes to the field by studying a few aspects of this class of systems. Topics covered include different timing schemes, network structures that sus-

tain cyclic dynamics, statistical complexity of large synchronous random networks, and the bias of a class of social game-inspired networks that incorporates memory effect and reciprocity.

Our studies of the autonomous Boolean network (ABN) model have elucidated many interesting features. One important feature is that a pulse cannot stably propagate around simple loops with an even number of NOT gates. These networks only host fixed-point attractors in the long term. On the other hand, loops with an odd number of NOT gates are forced into oscillations characterized by single kink propagation. We have also identified motifs that grow pulses and motifs that cuts down pulses. Combining these two types of motifs is a mechanism to achieve stable pulse oscillations. We find evidence that the yeast cell-cycle oscillator is embedded with such a combination. Our success in identifying functional network motifs and in building insights into the underlying logical structure of a biological system is not only testimony to the advantages of the delay update strategy, but also testimony to the applicability of Boolean modeling itself.

More generally speaking, the noisy autonomous Boolean network formalism presented in Chapter 3 provides a framework for studying the stability of network attractors in the Boolean regime. The autonomous Boolean network has become a subject of mathematical interest as well as a model of choice in studying real physical systems lately. One important mathematical question is whether there exists a mapping from the more general ODE model which has continuous state values to the autonomous Boolean model. While this is still an open question, Xianrui Cheng and Mengyang Sun of our research group, headed by Joshua E. S. Socolar, has found evidence of such a map in a particular group of ODE networks used in a model of drosophila cell division [81]. They binarized the continuous time series and extracted the effective time-delays from the ODEs, and confirmed that the ABN model reproduced the same pattern formation and transient behavior under certain conditions [58]. The added

advantage of the ABN model in this scenario is that timing constraints that leads to correct pattern formation can be derived without simulating the network dynamics, a convenience that the corresponding ODE model does not provide. Another exemplary use of the ABN model concerns the discovery of deterministic Boolean chaos. A synchronous Boolean network has a finite number of states and transitions between states are one-to-one, so its dynamics has to be periodic. When continuous time-delays are introduced to the Boolean model, it prompts us to ask the question whether chaos a possible behavior. Zhang, Cavalcante and colleagues observed ultra-wide-band power spectrum and positive Lyaponov exponents in simple networks of electronic logic gates that are not regulated by a clocking signal, and they discovered that an ABN model with short pulse rejection and history-dependent time delays is able to explain their observation. Their preliminary theoretical work also suggests that Boolean delay systems with only short pulse rejection and quenched time delays are not capable of chaotic behavior [57, 36].

Our investigation into the statistical complexity of large synchronous random Boolean networks (RBNs) is a study of RBN properties as well as a study of the complexity measures. We have found that when extending the formalism of Shalizi's complexity measure to RBNs, we must make a distinction between determining causal states by averaging over nodes at a given time step versus averaging over time for each node separately. The former strategy, originally adopted by Shalizi *et al.* in studying self-organization of cellular automata, when applied to RBNs, does not distinguish between the static spatial inhomogeneity of the ordered phase and the dynamical inhomogeneity of the disordered phase. The latter strategy, implemented by us, yields vanishing complexity values for networks in the ordered and critical regimes and for highly disordered networks, peaking somewhere in the disordered regime. This result is in agreement with the popular notion, first proposed by Crutchfield and Young [41], that complex dynamics lies between extreme orderness

and extreme randomness. We have demonstrated analytical insights in these complexity measures and obtained exact expressions in some cases. We have also seen evidence that individual nodes with have higher complexity are also the ones with a stronger influence to the dynamics of the whole network.

Our measure of complexity that treats individual network components as its own computational machinery can be readily applied on any heterogeneous extended systems. It would be very interesting to see our approach adopted in constructing computational models aimed at reproducing the statistical properties of time series from real systems, or in constructing algorithms aimed at picking out important components in large real systems. We note that the information theoretic study of complexity and related properties of discrete systems is becoming an active research area. For example, Wang *et al.* found that another complexity measure, Fisher information, is maximized at the critical phase of RBNs [82, 83]. Gershenson and Fernández defined emergence as the ratio between information inputs and outputs, self-organization as the difference between information inputs and outputs, and complexity as the product of emergence and self-organization [84]. The complexity data they show on $K-$regular random Boolean networks, whose distribution of logic function is defined by the a probability of having 1's in the logic output table, follows the same pattern in different dynamical phases as our measure. It is left for future work to determine the relations between these various complexity measures. It is a common sentiment in the community that clarity and agreement should be brought to the notions aimed at characterizing complex systems. The author of this dissertation echoes that sentiment, and believes that the study presented here contributes to the effort.

Throwing away vast amounts of microscopic details in order to get workable macroscopic descriptions is a well practiced tenet of statistical physics. Examining the level of details that can be ignored is an effort to understand the origin of emer-

gent system properties. The semi-annealed approximation technique presented in this dissertation is an example of finding the appropriate level of the microscopic details, in a mean field calculation, in order to produce satisfactory macroscopic statistics. Previous treatments of random networks found in the literature have largely avoided the presence of short loops, which are the primary source of local state correlations that render highly coarse-grained treatments less applicable. Our analytical tool considers the correlation between nodes and their immediate neighbors, and by doing so it has yielded significantly more accurate predictions of network bias in a class of social-game inspired networks where every node is in loops of lengths 1 and 2. Some interesting game theory related results are found as a byproduct of this study, and most of them are predicted by the semi-annealed approximation.

More network structures and logic function distributions should be studied with the semi-annealed approximation in the future, especially those with more relevance to social studies. Other dynamical measures, such as sensitivity to small perturbations, could be obtained with the new approximation scheme as well. Dynamical properties of deterministic random networks with topological features different from the traditional locally "tree-like" Kauffman networks have become a topic of recent interest. Other approximation methods recently developed can treat networks with features such as link assortativity, motifs, and community structures [73, 85]. The semi-annealed approximation developed here adds short loops to the list.

To me (the author), the seemingly loosely connected topics studied in this dissertation not only share a unifying theme of Boolean network dynamics, but also together represent a path of scientific endeavor guided by genuine curiosity in different fields. I have described new findings in gene regulatory networks, social networks, and dynamical systems theory. My own work has contributed most directly to the latter and at various levels indirectly to the others. It has been gratifying to participate and witness the uncovering of new pieces that fill in the gaps or extend our

current understanding, and it is with enthusiastic anticipation that I look forward to future discoveries.

Bibliography

[1] S. H. Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, March 2001.

[2] M. Newman. *Networks: An Introduction*. OUP Oxford, 2010.

[3] Adilson E. Motter and Reka Albert. Networks in motion. *Physics Today*, 65(4):43–48, 2012.

[4] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, June 1998.

[5] P. L. Krapivsky, S. Redner, and F. Leyvraz. Connectivity of growing random networks. *Phys. Rev. Lett.*, 85:4629–4632, Nov 2000.

[6] P. L. Krapivsky and S. Redner. Organization of growing random networks. *Phys. Rev. E*, 63:066123, May 2001.

[7] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS '00, Washington, DC, USA, 2000. IEEE Computer Society.

[8] Colin Cooper and Alan Frieze. A general model of web graphs. *Random Structures and Algorithms*, 22(3):311–335, 2003.

[9] Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science*, 286(5439):509–512, October 1999.

[10] S.A. and Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437 – 467, 1969.

[11] Stuart Kauffman. The large scale structure and dynamics of gene control circuits: An ensemble approach. *Journal of Theoretical Biology*, 44(1):167 – 190, 1974.

[12] Stuart A. Kauffman. Emergent properties in random complex automata. *Physica D: Nonlinear Phenomena*, 10(12):145 – 156, 1984.

[13] B. Derrida and Y. Pomeau. Random networks of automata: A simple annealed approximation. *EPL (Europhysics Letters)*, 1(2):45, 1986.

[14] Juha Kesseli, Pauli Rämö, and Olli Yli-Harja. Iterated maps for annealed boolean networks. *Phys. Rev. E*, 74:046104, Oct 2006.

[15] Richard Durrett, James P. Gleeson, Alun L. Lloyd, Peter J. Mucha, Feng Shi, David Sivakoff, Joshua E. S. Socolar, and Chris Varghese. Graph fission in an evolving voter model. *Proceedings of the National Academy of Sciences*, 109(10):3682–3687, March 2012.

[16] S. Wolfram. *Cellular automata and complexity: collected papers*. Addison-Wesley Pub. Co., 1994.

[17] Cosma Rohilla Shalizi, Kristina Lisa Shalizi, and Robert Haslinger. Quantifying self-organization with optimal predictors. *Phys. Rev. Lett.*, 93:118701, Sep 2004.

[18] Xinwei Gong and Joshua E. S. Socolar. Quantifying the complexity of random boolean networks. *Phys. Rev. E*, 85:066107, Jun 2012.

[19] Uri Alon. Network motifs: theory and experimental approaches. *Nat Rev Genet*, 8(6):450–461, June 2007.

[20] Volkan Sevim, Xinwei Gong, and Joshua E. S. Socolar. Reliability of transcriptional cycles and the yeast cell-cycle oscillator. *PLoS Comput Biol*, 6(7):e1000842, 07 2010.

[21] I. Harvey and T. Bossomaier. Time out of joint: Attractors in asynchronous random boolean networks. In *Proceedings of the Fourth European Conference on Artificial Life*, pages 67–75. MIT Press, Cambridge, 1997.

[22] E.A. Di Paolo. Rhythmic and non-rhythmic attractors in asynchronous random boolean networks. *BioSystems*, 59(3):185–195, 2001.

[23] C. Gershenson. Classification of random boolean networks. In *Proc. of the 8th Int. Conf. on Artificial Life*, pages 1–8, 2002.

[24] Stuart A. Kauffman. *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, USA, 1 edition, June 1993.

[25] Barbara Drossel, Tamara Mihaljev, and Florian Greil. Number and length of attractors in a critical kauffman model with connectivity one. *Phys. Rev. Lett.*, 94:088701, Mar 2005.

[26] Andre S. Ribeiro, Stuart A. Kauffman, Jason Lloyd-Price, Björn Samuelsson, and Joshua E. S. Socolar. Mutual information in random boolean models of regulatory networks. *Phys. Rev. E*, 77:011901, Jan 2008.

[27] Stephen E. Harris, Bruce K. Sawhill, Andrew Wuensche, and Stuart Kauffman. A model of transcriptional regulatory networks based on biases in the observed regulation rules. *Complex.*, 7(4):23–40, March 2002.

[28] Pauli Rämö, Juha Kesseli, and Olli Yli-Harja. Stability of functions in boolean models of gene regulatory networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 15(3):034101, 2005.

[29] L. Kadanoff, S. Coppersmith, and M. Aldana. Boolean Dynamics with Random Couplings. *eprint arXiv:nlin/0204062*, April 2002.

[30] R.J. Bagley and Leon Glass. Counting and classifying attractors in high dimensional dynamical systems. *Journal of Theoretical Biology*, 183(3):269 – 284, 1996.

[31] U. Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman and Hall/CRC mathematical & computational biology series. Chapman & Hall/CRC, 2007.

[32] Johannes Norrell and Joshua E. S. Socolar. Boolean modeling of collective effects in complex networks. *Phys. Rev. E*, 79:061908, Jun 2009.

[33] D Dee and M Ghil. Boolean difference equations. i - formulation and dynamic behavior. *SIAM Journal on Applied Mathematics*, 44(1):111, 1984.

[34] M. Ghil and A. Mullhaupt. Boolean delay equations. ii. periodic and aperiodic solutions. *Journal of Statistical Physics*, 41:125–173, 1985. 10.1007/BF01020607.

[35] Michael Ghil, Ilya Zaliapin, and Barbara Coluzzi. Boolean delay equations: A simple way of looking at complex systems. *Physica D: Nonlinear Phenomena*, 237(23):2967 – 2986, 2008.

[36] Rui Zhang, Hugo L. D. de S.Cavalcante, Zheng Gao, Daniel J. Gauthier, Joshua E. S. Socolar, Matthew M. Adams, and Daniel P. Lathrop. Boolean chaos. *Phys. Rev. E*, 80:045202, Oct 2009.

[37] Jeff Hasty, Joel Pradines, Milos Dolnik, and J. J. Collins. Noise-based switches and amplifiers for gene expression. *Proceedings of the National Academy of Sciences*, 97(5):2075–2080, February 2000.

[38] Konstantin Klemm and Stefan Bornholdt. Stable and unstable attractors in boolean networks. *Phys. Rev. E*, 72:055101, Nov 2005.

[39] C. E. Shannon. A mathematical theory of communication. *Bell system technical journal*, 27, 1948.

[40] James P. Crutchfield. Between order and chaos. *Nat Phys*, 8(1):17–24, January 2012.

[41] James P. Crutchfield and Karl Young. Inferring statistical complexity. *Phys. Rev. Lett.*, 63:105–108, Jul 1989.

[42] Cosma Rohilla Shalizi. *Causal Architecture, Complexity and Self-Organization in Time Series and Cellular Automata*. PhD thesis, University of Wisconsin at Madison, May 2001.

[43] Cosma Rohilla Shalizi and James P. Crutchfield. Computational mechanics: Pattern and prediction, structure and simplicity. *Journal of Statistical Physics*, 104:817–879, 2001. 10.1023/A:1010388907793.

[44] Cosma Rohilla Shalizi. Optimal nonlinear prediction of random fields on networks. In Michel Morvan and Éric Rémila, editors, *Discrete Models for Complex Systems, DMCS'03*, volume AB of *DMTCS Proceedings*, pages 11–30. Discrete Mathematics and Theoretical Computer Science, 2003.

[45] S. Kullback. *Information theory and statistics*. Dover books on mathematics. Dover Publications, 1997.

[46] Julio E Celis, Mogens Kruhffer, Irina Gromova, Casper Frederiksen, Morten stergaard, Thomas Thykjaer, Pavel Gromov, Jinsheng Yu, Hildur Plsdttir, Nils Magnusson, and Torben F rntoft. Gene expression profiling: monitoring transcription and translation products using dna microarrays and proteomics. *FEBS Letters*, 480(1):2 – 16, 2000.

[47] D. J. Lockhart and E. A. Winzeler. Genomics, gene expression and DNA arrays. *Nature*, 405(6788):827–836, June 2000.

[48] Ilya Shmulevich and Wei Zhang. Binary analysis and optimization-based normalization of gene expression data. *Bioinformatics*, 18(4):555–565, 2002.

[49] John J Tyson, Katherine C Chen, and Bela Novak. Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Current Opinion in Cell Biology*, 15(2):221 – 231, 2003.

[50] J.D. Murray. *Mathematical Biology*. Number v. 2 in Interdisciplinary applied mathematics. Springer, 2002.

[51] Michael B. Elowitz and Stanislas Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, January 2000.

[52] John J. Tyson, Reka Albert, Albert Goldbeter, Peter Ruoff, and Jill Sible. Biological switches and clocks. *Journal of the Royal Society, Interface / the Royal Society*, 5 Suppl 1(Suppl 1):S1–S8, August 2008.

[53] Andrew W Murray. Recycling the cell cycle: Cyclins revisited. *Cell*, 116(2):221 – 234, 2004.

[54] S. B. Haase and S. I. Reed. Evidence that a free-running oscillator drives G1 events in the budding yeast cell cycle. *Nature*, 401(6751):394–397, September 1999.

[55] David A. Orlando, Charles Y. Lin, Allister Bernard, Jean Y. Wang, Joshua E. S. Socolar, Edwin S. Iversen, Alexander J. Hartemink, and Steven B. Haase. Global control of cell-cycle transcription by coupled CDK and network oscillators. *Nature*, 453(7197):944–947, May 2008.

[56] Fangting Li, Tao Long, Ying Lu, Qi Ouyang, and Chao Tang. The yeast cell-cycle network is robustly designed. *Proceedings of the National Academy of Sciences of the United States of America*, 101(14):4781–4786, April 2004.

[57] Hugo L. D. de S. Cavalcante, Daniel J. Gauthier, Joshua E. S. Socolar, and Rui Zhang. On the origin of chaos in autonomous booleannetworks. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1911):495–513, 2010.

[58] Xianrui Cheng, Mengyang Sun, and Joshua E. S. Socolar. Autonomous boolean modeling of developmental gene regulatory networks. 2012.

[59] Peter Grassberger. Toward a quantitative theory of self-generated complexity. *International Journal of Theoretical Physics*, 25:907–938, 1986. 10.1007/BF00668821.

[60] Barbara Drossel. *Random Boolean Networks*, pages 69–110. Wiley-VCH Verlag GmbH & Co. KGaA, 2009.

[61] Björn Samuelsson and Joshua E. S. Socolar. Exhaustive percolation on random networks. *Phys. Rev. E*, 74:036113, Sep 2006.

[62] M. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[63] J. Scott. *Social Network Analysis: A Handbook*. Sage Publications, 2000.

[64] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Structural Analysis in the Social Sciences. Cambridge University Press, 1994.

[65] Paul W. Holland and Samuel Leinhardt. Transitivity in structural models of small groups. *Comparative Group Studies*, 2:107–124, 1971.

[66] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, Jun 1998.

[67] M. E. J. Newman and Juyong Park. Why social networks are different from other types of networks. *Phys. Rev. E*, 68:036122, Sep 2003.

[68] James Moody. Race, school integration, and friendship segregation in america. *American Journal of Sociology*, 107(3):pp. 679–716, 2001.

[69] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002.

[70] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences of the United States of America*, 98(2):404–409, January 2001.

[71] S.N. Coppersmith, Leo P. Kadanoff, and Zhitong Zhang. Reversible boolean networks i: distribution of cycle lengths. *Physica D: Nonlinear Phenomena*, 149(12):11 – 29, 2001.

[72] J. A. de Sales, M. L. Martins, and D. A. Stariolo. Cellular automata model for gene networks. *Phys. Rev. E*, 55:3262–3270, Mar 1997.

[73] Andrew Pomerance, Edward Ott, Michelle Girvan, and Wolfgang Losert. The effect of network topology on the stability of discrete state models of genetic control. *Proceedings of the National Academy of Sciences*, 106(20):8209–8214, May 2009.

[74] John von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, IL, USA, 1966. Von Neumann's work on self-reproducing automata, completed and edited after his death by Arthur Burks.

[75] Stephen Wolfram. Statistical mechanics of cellular automata. *Rev. Mod. Phys.*, 55:601–644, Jul 1983.

[76] S. Wolfram. *A new kind of science*. General science. Wolfram Media, 2002.

[77] S. Wolfram. *Theory and Applications of Cellular Automata: Including Selected Papers, 1983-1986*. Advanced Series on Complex Systems. World Scientific, 1986.

[78] Martin A. Nowak and Robert M. May. The sptial dilemmas of evolution. *Int J Bifurcat Chaos*, 3:35–78, 1993.

[79] Vincent Marceau, Pierre-André Noël, Laurent Hébert-Dufresne, Antoine Allard, and Louis J. Dubé. Adaptive networks: Coevolution of disease and topology. *Phys. Rev. E*, 82:036116, Sep 2010.

[80] James P. Gleeson. High-accuracy approximation of binary-state dynamics on networks. *Phys. Rev. Lett.*, 107:068701, Aug 2011.

[81] Nicholas T Ingolia. Topology and robustness in the *Drosophila* segment polarity network. *PLoS Biol*, 2(6):e123, 06 2004.

[82] X. Rosalind Wang, Joseph T. Lizier, and Mikhail Prokopenko. Fisher information at the edge of chaos in random Boolean networks. *Artificial Life*, 17(4):315–329, 2011.

[83] Mikhail Prokopenko, Joseph T. Lizier, Oliver Obst, and X. Rosalind Wang. Relating fisher information to order parameters. *Phys. Rev. E*, 84:041116, Oct 2011.

[84] Carlos Gershenson and Nelson Fernandez. Complexity and information: Measuring emergence, self-organization, and homeostasis at multiple scales. *CoRR*, abs/1205.2026, 2012.

[85] S. Squires, E. Ott, and M. Girvan. Dynamical Instability in Boolean Networks as a Percolation Problem. *ArXiv e-prints*, January 2012.

Biography

Xinwei Gong, commonly known as Sam, was born on May 22, 1984 in Hunan, China. He spent most of his childhood and adolescent years in the southern Chinese city of Shenzhen.

After graduating from Shenzhen Experimental School in July 2002, Sam matriculated into Virginia Polytechnic Institute and State University (Virginia Tech) and majored in physics and mathematics. He participated in numerous research projects and was twice awarded the Robert C. Richardson scholarship for his academic achievements. In May 2006, Sam received dual B.S. degrees with honors in physics and mathematics.

In July 2006, Sam started graduate studies in physics at Duke University with advisor Joshua Socolar. At Duke, Sam took an interest in teaching, and had the opportunity to lecture in graduate and undergraduate courses. He was once awarded the graduate teaching fellowship. Sam's research trajectory at Duke was heavily influenced by Duke's interdisciplinary culture. With academic and financial support from the Center for Nonlinear and Complex Systems, Duke Center for Systems Biology, and Duke Network Analysis center, Sam's research work led to publications and culminated in this dissertation. Upon successful defense, he will graduate with a Ph.D. in physics in summer 2012.

Sam has accepted a senior scientist position in a biotechnology firm in Palo Alto, California. He is expected to begin work in Aug. 2012.