Symbolic dynamics and computation in model gene networks

R. Edwards[a)]
Department of Mathematics and Statistics, University of Victoria, P. O. Box 3045, STN CSC, Victoria, British Columbia, Canada V8W 3P4

H. T. Siegelmann[b)]
Information Systems Engineering, Faculty of Industrial Engineering, Technion 32000 Haifa, Israel

K. Aziza and L. Glass[c)]
Centre for Nonlinear Dynamics in Physiology and Medicine, Department of Physiology, McIntyre Medical Sciences Building, McGill University, 3655 Drummond St., Montréal, Québec, Canada H3G 1Y6

(Received 17 July 2000; accepted for publication 30 October 2000)

We analyze a class of ordinary differential equations representing a simplified model of a genetic network. In this network, the model genes control the production rates of other genes by a logical function. The dynamics in these equations are represented by a directed graph on an n-dimensional hypercube (n-cube) in which each edge is directed in a unique orientation. The vertices of the n-cube correspond to orthants of state space, and the edges correspond to boundaries between adjacent orthants. The dynamics in these equations can be represented symbolically. Starting from a point on the boundary between neighboring orthants, the equation is integrated until the boundary is crossed for a second time. Each different cycle, corresponding to a different sequence of orthants that are traversed during the integration of the equation always starting on a boundary and ending the first time that same boundary is reached, generates a different letter of the alphabet. A word consists of a sequence of letters corresponding to a possible sequence of orthants that arise from integration of the equation starting and ending on the same boundary. The union of the words defines the language. Letters and words correspond to analytically computable Poincaré maps of the equation. This formalism allows us to define bifurcations of chaotic dynamics of the differential equation that correspond to changes in the associated language. Qualitative knowledge about the dynamics found by integrating the equation can be used to help solve the inverse problem of determining the underlying network generating the dynamics. This work places the study of dynamics in genetic networks in a context comprising both nonlinear dynamics and the theory of computation. [DOI: 10.1063/1.1336498]

Networks of genes underlie the normal development and function of organisms. Information about the structure of the genome of humans and other organisms is increasing exponentially. However, the ways in which the genes regulate and control behavior are still not well understood. A simple mathematical model is discussed in which outputs from a gene act to control and regulate the activity of other genes in the network. The interactions can be represented as simple logical rules. However, the resulting dynamics can be quite complex, even in simple networks composed of only four model genes. Therefore, it is useful to adopt symbolic methods to describe the dynamics. The symbolic methods represent complicated dynamics by strings of symbols, and a correspondence can be established with logical automata that generate similar symbolic strings. Changes in parameters in the simple model can elicit changes in the symbolic sequences, corresponding to bifurcations between different patterns of chaotic dynamics. Thus, the gene networks can be thought of as computational devices that generate languages. Further, based on observation of qualitative properties of the dynamics, represented by the levels of activities of genes and whether their products are increasing or decreasing, it is possible to devise methods to carry out the inverse problem—i.e., to determine the underlying logical network generating the observed dynamics. This work places the study of dynamics in gene networks in a computational perspective and may lead to new methods to study the functional properties in gene networks.

I. INTRODUCTION

Recent years have witnessed exponential increases in our knowledge about the sequences of nucleotides in the genomes of living organisms. In addition, gene expression chips now enable scientists to monitor activity levels of thousands of genes simultaneously.[1,2] These experimental advances are leading us into a new era in which the overriding questions will involve understanding the mechanisms that regulate gene expression, and lead to the coordinated function of multiple genes. The current work is based on the following premises: (i) it is sensible to think about genes as being switched ''on'' or ''off;'' (ii) crude knowledge about

[a)]Electronic mail: edwards@math.uvic.ca
[b)]Electronic mail: iehava@ie.technion.ac.il
[c)]Electronic mail: glass@cnd.mcgill.ca

the interactions among genes may suffice to understand the functioning of gene networks; (iii) a simplified mathematical framework may be suitable to capture a variety of qualitative features of genetic networks that are relevant to their functional and computational abilities; and (iv) symbolic dynamic approaches to dynamical systems[3,4] and analysis of languages in the theory of computation[5–7] form a natural bridge to consider the dynamical and computational properties of differential equations modeling genetic networks.

This work is rooted in early studies by McCulloch and Pitts, who proposed that binary switching devices operating in discrete time could be used to model neural networks.[8] They showed that while networks composed of a finite number of such neurons are computationally equivalent to finite state machines, a countable number of these neurons had potentially the power of a Turing machine. Largely inspired by the original McCulloch and Pitts model of the neuron, Kauffman proposed that genetic networks could be modeled by random Boolean networks in which time is discrete and each element computes a Boolean function based on the values of inputs to that element.[9] In contrast to the work on neural networks, in which emphasis was placed on the computational properties,[10,11] in Kauffman's analysis of genetic networks, emphasis was placed on dynamic aspects. Steady states and cycles in the logical network were equated with differentiated cell types in the organism and a variety of extensions have been explored.[12,13]

Since gene networks do not act in discrete time and gene product concentrations are continuous variables, we believe that the discrete networks above, or even asynchronous versions of them, are less suitable to model gene networks than ordinary differential equations in which gene interactions are incorporated as logical functions.[14–22] Differential equations and logical networks have been proposed to model a variety of different specific gene networks.[23–27]

In this paper we analyze genetic network models both in terms of computational capabilities and in terms of dynamical properties. This combination should provide an interesting bridge between computer science and dynamical systems.

II. A DIFFERENTIAL EQUATION

In this section we briefly present a mathematical model of gene networks. Since many aspects of the model have recently been reviewed,[21,22] we refer the reader to these earlier publications for further mathematical details.

A Boolean switching network with N elements is represented

$$X_i(j+1)=\Lambda_i(X_{i_1}(j),X_{i_2}(j),\ldots,X_{i_K}(j)),\qquad i=1,\ldots,N, \tag{1}$$

where $\Lambda_i(X_{i_1}(j),X_{i_2}(j),\ldots,X_{i_K}(j))\in\{0,1\}$ and K is the number of inputs. This is a discrete time and discrete state space system. Therefore, it must eventually reach a fixed point or cycle under iteration.

Since biological systems are not believed to have clocking devices that simultaneously update the network, a differential equation would be a more suitable class of mathematical model. The logical structure of Eq. (1) can be captured by a differential equation.[15,16,18] To a continuous variable $x_i(t)$, we associate a discrete variable $X_i(t)$,

$$X_i(t)=0 \quad \text{if } x_i(t)<0; \quad \text{otherwise, } X_i(t)=1. \tag{2}$$

For any logical network, we define an analogous differential equation,

$$\frac{dx_i}{dt}=-x_i+\lambda_i(X_{i_1}(j),X_{i_2}(j),\ldots,X_{i_K}(j)),\qquad i=1,\ldots,N, \tag{3}$$

where $\lambda_i(X_{i_1}(j),X_{i_2}(j),\ldots,X_{i_K}(j))$ is a scalar whose sign is negative (positive) if the corresponding logical variable $\Lambda_i(X_{i1}(j),X_{i2}(j),\ldots,X_{i_K}(j))$ is 0 (1).

For each variable, the temporal evolution is governed by a first order piecewise linear differential equation. Let $\{t_1,t_2,\ldots,t_k\}$, denote the *switch times* when any variable of the network crosses 0. The solution of Eq. (3) for each variable x_i for $t_j<t<t_{j+1}$, is

$$x_i(t)=x_i(t_j)\,e^{-(t-t_j)}+\lambda_i(X_{i_1}(j),X_{i_2}(j),\ldots,X_{i_K}(j))(1-e^{-(t-t_j)}). \tag{4}$$

This equation has the following property. All trajectories in a given orthant in state space are directed towards a focal point. If the focal point lies in a different orthant from the initial condition, then, in general, eventually a threshold hyperplane will be crossed. When the threshold hyperplane is crossed, a new focal point may be selected based on the underlying equations of motion.

Even though Eq. (3) is more realistic than Eq. (1) as a model for biological systems, this equation still is a highly oversimplified model for real systems. Yet this equation has remarkable mathematical properties that facilitate theoretical analysis. Moreover, there is an expectation, demonstrated in some simple examples, that the qualitative dynamics in the model system will be preserved in more realistic versions, for example, when the discontinuous step functions are replaced by continuous sigmoidal functions.[17]

In the differential equation, in general only one variable will cross its threshold at a given time. Therefore, the dynamics in the differential equation can be mapped on an N-cube where directed edges represent allowed transition between logical states. The allowed transitions are also equivalent to the allowed transitions in an asynchronous switching network with the same logical structure.[12,16,18]

Further, for networks in which there is no self-input, each edge of the N-cube representation will have a unique orientation.[16,18] If the network has self-input, there may occur black walls or white walls,[19] threshold hyperplanes for which nearby trajectories approach or retreat (respectively) from both sides. These can be represented on the N-cube by a pair of arrows pointing inward from each end of the edge, or outward, respectively. Self-input may in some cases be an appropriate description of autocatalysis, wherein a gene's protein product either represses or activates its own synthesis, as in the case of viral genes (cI and cro) in bacteriophage

TABLE I. Truth table for a four-variable Boolean network. This defines the mapping for a discrete-time network as well as the signs of the interaction terms in the differential equations for continuous-time networks [in this case, w_{t+1}, etc., should be interpreted rather as the sign of $(dw/dt)+w$].

$(xy)_t$	w_{t+1}	$(wz)_t$	x_{t+1}	$(wx)_t$	y_{t+1}	$(xy)_t$	z_{t+1}
(00)	0	(00)	1	(00)	1	(00)	1
(01)	1	(01)	0	(01)	0	(01)	1
(10)	1	(10)	0	(10)	0	(10)	0
(11)	0	(11)	1	(11)	0	(11)	1

lambda.[24,28] We will primarily be concerned here with networks with no self-input, though this is by no means essential. We define two networks to belong to the same *structural equivalence class* if their directed *N*-cube representations are identical under a symmetry operation of the *N*-cube. However, since changes in the location of the focal points $\{\lambda_i\}$, can lead to bifurcations in the dynamics even though the directed graph representation is unchanged, the structural equivalence class is not necessarily sufficient to specify the dynamics.

In some cases, the *N*-cube mapping gives precise information about the qualitative dynamics of its associated differential equation. A vertex on the *N*-cube with only edges directed towards it corresponds to a stable steady state in the differential equation. A cycle on the *N*-cube is attracting if for each vertex on the cycle, each of the $N-2$ adjacent vertices not on the cycle are directed towards it. An attracting cycle on the *N*-cube will be associated with either a stable limit cycle or a stable focus in the ordinary differential equations.[18] Chaotic dynamics arises in systems with multiple cycles passing through individual vertices on the *N*-cube but no firm results allow us to identify which differential equations admit chaos based on the *N*-cube mapping.

We illustrate these ideas with a four-dimensional network that displays chaotic dynamics.[20] The network structure is defined by the truth tables for each variable in Table I. This is equivalent to the single truth table in Table II.

TABLE II. The combined truth table for the Boolean network in Table I. Some of the associated differential equations with this sign structure give chaotic dynamics.

$(wxyz)_t$	$(wxyz)_{t+1}$
(0000)	0111
(0001)	0011
(0010)	1111
(0011)	1011
(0100)	1100
(0101)	1000
(0110)	0101
(0111)	0001
(1000)	0001
(1001)	0101
(1010)	1001
(1011)	1101
(1100)	1000
(1101)	1100
(1110)	0001
(1111)	0101

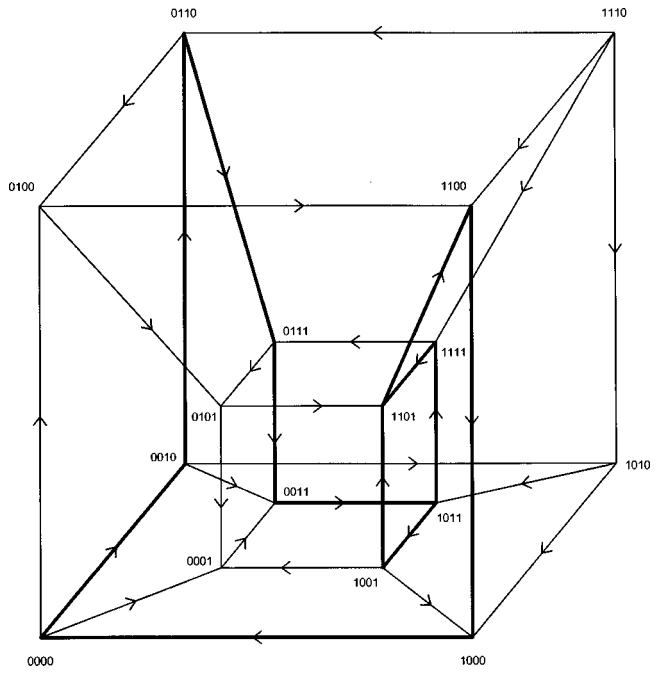

FIG. 1. 4-cube representation of a switching network. An associated differential equation, Eq. (5), displays chaotic dynamics. The bold lines correspond to the two cycles $\mathcal{A}$ and $\mathcal{B}$ that occur on the chaotic attractor. Based on Fig. 1 in Mestl *et al.* (Ref. 20).

The directed graph on the *N*-cube is generated by drawing arrows from each vertex in the left-hand column to all adjacent vertices (i.e., vertices that lie on a Hamming distance of 1 away from the vertex in the left-hand column) that lie on a shortest route from that vertex to the vertex in the right-hand column. The number of directed edges emanating from a given vertex is equal to the Hamming distance between that state in the truth table at time (t) and the target state at time $(t+1)$. For example, there are three edges directed out from 0000 towards 0001, 0010, and 0100. The *N*-cube representation for this network is shown in Fig. 1.

The four-dimensional differential equation,

$$\begin{aligned}
\frac{dx_1}{dt} &= 2(X_2\overline{X}_3+\overline{X}_2X_3)-c-x_1,\\
\frac{dx_2}{dt} &= 2(\overline{X}_1\overline{X}_4+X_1X_4)-1.3762-x_2,\\
\frac{dx_3}{dt} &= 2X_1X_2-0.8024-x_3,\\
\frac{dx_4}{dt} &= 2(\overline{X}_1\overline{X}_3+X_3)-1.2682-x_4,
\end{aligned} \tag{5}$$

where $\overline{X}=1-X$, and c is a constant to be selected is consistent with the transition diagram in Fig. 1; when $c=1.2546$ this equation displays chaos.[20] We return to this example after developing some additional terminology.

III. SYMBOLIC DYNAMICS OF GENETIC NETWORKS

Symbolic methods have provided powerful techniques for analysis of dynamical systems.[3] Since symbolic methods

are discrete, they are also amenable to analysis using the theory of computation[6] and in particular the theory of automata.[5] Several works have explored the interface between symbolic dynamic representations of nonlinear dynamical systems and computation.[4,7,29,30] Here we choose a particular definition of symbolic dynamics based on the Poincaré section, and show how this symbolic dynamics provides a link between the qualitative dynamics of Eq. (3) and the theory of computation, thus enabling computational interpretation of this dynamical system.

We consider Eq. (3) for the situation in which the dynamics do not approach a stable fixed point. Therefore, the dynamics are either periodic, quasiperiodic, or chaotic.

For any particular equation of the form of Eq. (3), we associate a generative finite state machine and a formal language. Since there are various definitions of these terms, we concisely describe our notation.

A. Definitions and notation

A *finite state machine* is a tuple $\mathcal{M}=(Q,q_0,\Sigma_i,\Sigma_o,E)$ where:

(1) Q is a finite set of states having one designated state: q_0 is the initial state;
(2) Σ_i and Σ_o are two sets of letters called the input alphabet and the output alphabet, respectively; and
(3) E is the partial transition function $E:Q\rightarrow Q$. Each edge is associated with a set of letters from Σ_i and a string of (0 or more) letters from Σ_o.

The finite state machine can be represented as a graph in which each state is a node and the directed edges between nodes represent the partial transition function. Each traverse of the automaton along the directed edges generates a *word* which is the concatenation of the strings of output alphabet associated with the traversed edges. In this definition of a traverse, it has no designated end, and hence it can be stopped or continue forever. The associated words can thus be finite and infinite. The set of such words for all the various inputs constitutes the *formal language* associated with the (generative) state machine. In the theory of computation, the sets of alphabet letters are finite and typically consist of two letters only. In the model of computation over the real numbers,[7] the input alphabet is the infinite set of real values **R**, and the output alphabet is finite. Here, we allow for both sets to be of any size.

Languages associated with finite state machines having finite alphabet sets are called *regular*. We follow the notation of Ref. 5, p.28: let Σ be an alphabet, and L,L_1,L_2 be sets of words on Σ. $L_1L_2=\{xy|\ x\in L_1,y\in L_2\}$, $L^0=\{\epsilon\}$, $L^i=LL^{i-1}$ for $i\geqslant 1$, $L^*=\cup_{i=0}^{\infty}L^i$, $L^+=\cup_{i=1}^{\infty}L^i$. The empty set having no word $\{\ \}$ and the singleton set including only the empty word $\{\epsilon\}$ are both regular languages. If L_1 and L_2 are regular languages, so are $L_1\cup L_2$, L_1L_2, L_1^*, and L_1^+.

B. Associating a language with Eq. (3)

There are various ways to associate languages with the dynamics of Eq. (3). We first propose a conceptually simple method that offers a clear connection between the dynamics and formal languages.

We assume that all transients have passed and select an edge e_0 that is traversed during the dynamics. Then we integrate Eq. (3) until the edge is traversed again *for the first time*. The sequence of edges of the graph that were traversed define a cycle, that we associate with a symbol. Each different cycle, that starts and ends on e_0 when that edge is first encountered, generates a new symbol. The set of symbols so generated defines the output alphabet. The set of all strings representing paths starting at initial conditions from e_0 defines the language $\mathcal{L}(e_0)$. These definitions generate an intimate connection between languages and dynamics. Edges on the N-cube correspond to boundaries between orthants of phase space, so a cycle of edges corresponds to a return map on an orthant boundary. The letters constituting the output alphabet thus correspond to (analytically or numerically computable) return maps and the words correspond to compositions of these return maps.

The hypercube dynamics do not describe completely the underlying dynamical system. The hypercubes seem nondeterministic (because there can be more than one outgoing edge from a node) while the underlying dynamics are deterministic, and the choice of outgoing edge depends on the exact location in phase space of a trajectory as it crosses the orthant boundary corresponding to a given edge. For an exact formulation as a language, the hypercube needs an input alphabet in addition to the output alphabet. The input alphabet is the real value that represents the exact position in phase space of the system on the incoming orthant boundary. We think of this input alphabet as a hidden input, but must remember it is there for the deterministic computation.

We now consider an example. In Eq. (5) for $c=1.2546$ there is chaos.[20] Starting on the edge between 0011 and 1011, we encounter the following vertices in defining two cycles:

$$\mathcal{A}:1011\rightarrow\mathbf{1111}\rightarrow 1101\rightarrow 1100\rightarrow 1000\rightarrow 0000\rightarrow 0010\rightarrow 0110\rightarrow 0111\rightarrow 0011\rightarrow 1011,$$

$$\mathcal{B}:1011\rightarrow\mathbf{1001}\rightarrow 1101\rightarrow 1100\rightarrow 1000\rightarrow 0000\rightarrow 0010\rightarrow 0110\rightarrow 0111\rightarrow 0011\rightarrow 1011.$$

The vertex in bold is the only vertex that differs between the two cycles. From numerical integration we find that the symbol $\mathcal{A}$ can appear any number of times in sequence, but the symbol $\mathcal{B}$ only appears singly. Then the language for this equation is $\mathcal{B}^k(\mathcal{A}^+\mathcal{B})^*$, where k is either 0 or 1 and we take $\mathcal{B}^0\doteq\epsilon$. The first term arises because the first symbol generated may be $\mathcal{B}$.

This language has been called the *golden mean shift* by Lind and Marcus.[4] This language can also be generated by the finite deterministic automaton diagrammed in Fig. 2 called the *golden mean machine* by Crutchfield.[29]

In this case the alphabet is finite (two symbols). However, had we started on the edge between 1011 and 1001 the language generated by the network would have been different. Now each cycle could loop an arbitrary number of times around the $\mathcal{A}$ loop defined above before returning to the edge between 1011 and 1001. Thus, the alphabet here is now infinite. This observation underscores both a weakness and a

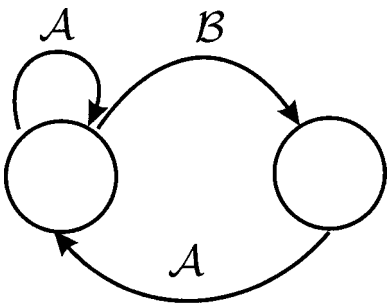

FIG. 2. The finite state machine generating the language associated with chaotic dynamics in Eq. (5) with parameter $c=1.2546$.

strength of our formulation. Although it seems artificial to generate different alphabets depending on the initial edge, we propose that different languages associated with exactly the same attractor in the differential equation (3) be considered ''in agreement.''

Although the association of the alphabet with return maps to a hyperplane has a natural dynamical interpretation, other conventions for associating languages with dynamics can also be adopted.

For example, in the example considered above we might define languages based on sequences of vertices traversed during the dynamics. Starting at vertex 1011 we can define a language with three letters:

$$A{:}1011\rightarrow\mathbf{1111}\rightarrow1101,$$

$$B{:}1011\rightarrow\mathbf{1001}\rightarrow1101,$$

$$C{:}1101\rightarrow1100\rightarrow1000\rightarrow0000\rightarrow0010\rightarrow0110\rightarrow0111\rightarrow0011\rightarrow1011.$$

The language is now $(BC)^k((AC)^+BC)^*$, where $k=0,1$. The structure of the automaton is as before (Fig. 2) but the edges that were associated with $\mathcal{A}$ and $\mathcal{B}$ are now associated with AC and BC, respectively. Breaking up the cycles in this way can eliminate the possibility of infinite alphabets.

The N-cube representation of a differential equation provides a way to classify networks in structural equivalence classes based on the symmetries of the N-cube. Symbolic dynamics provides a basis for discussing additional types of equivalence for Eq. (3) based on qualitative features of the dynamics. Two networks are in the same *dynamical equivalence class* if the languages associated with each network's attractors are identical, except for perhaps a fixed number of initial letters. This latter exception allows one to concentrate on the long-term dynamics without much emphasis on the initial state. For example, consider a differential equation of the form of Eq. (3) (in any dimension), whose only attractor is a stable limit cycle in which each edge associated with the limit cycle is traversed only once. Then, independent of the chosen initial state, the language associated with the differential equation will be $\mathcal{A}^*$ (after possibly a few initial letters). Thus, all dynamical systems of the form of Eq. (3) that converge to a simple globally attracting limit cycle, are in the same dynamical equivalence class using this definition. Note, however, that a network may have multiple attractors.

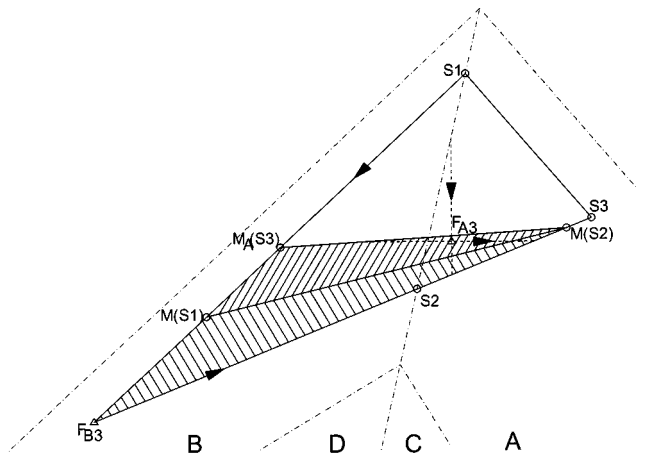

FIG. 3. Sketch of a trapping region in the $(0,-,+,+)$ boundary, for the network of Eq. (5) with $c=1.2546$, projected onto a plane orthogonal to vector $F_{\mathcal{A}3}$, an unstable fixed point of cycle $\mathcal{A}$. This and the following two figures are distorted to make the relevant regions visible. Based on Fig. 4(b) in Mestl *et al.* (Ref. 20).

IV. SYMBOLIC DYNAMICS AND BIFURCATIONS OF CHAOTIC DYNAMICS

One of the central questions in the field of nonlinear dynamics is to determine the changes in qualitative dynamics (i.e., the number, type, and stability of invariant sets) of dynamical systems as the parameters in those systems change. The symbolic representation of dynamics in Eq. (3) provides a new method to represent qualitative aspects of the dynamics. Under changes of the focal points, there can be changes in the associated language.

We illustrate these ideas by returning to Eq. (5) and consider the dynamics that are observed as c is changed. Most of the values of c given below can in principle be calculated exactly, using exact calculations of the return maps and their fixed points[19,21] with the parameter c left unspecified, and then solving for c under the desired condition. In practice, it is easier to do exact calculations with particular values of c and locate a bifurcation point by trying different values of c on either side. We denote the return maps for the $\mathcal{A}$ and $\mathcal{B}$ cycles on the $(0,-,+,+)$ boundary by $M_{\mathcal{A}}$ and $M_{\mathcal{B}}$, respectively. Composite mappings we denote by, for example, $M_{\mathcal{BA}}(x)\doteq M_{\mathcal{A}}(M_{\mathcal{B}}(x))$. When x is on the boundary between the $\mathcal{A}$ and $\mathcal{B}$ domains, the two mappings are equivalent so in this case we sometimes use $M(x)=M_{\mathcal{A}}(x)=M_{\mathcal{B}}(x)$.

When $c=1.2546$, there is a trapping region in the $(0,-,+,+)$ orthant boundary,[20] shown schematically in Fig. 3. $F_{\mathcal{B}3}$ is an unstable fixed point of the $\mathcal{B}$ cycle. $F_{\mathcal{A}3}$ is an unstable fixed point of the $\mathcal{A}$ cycle. The stable and unstable manifolds of these points are indicated by the arrows. In the figure, everything is projected onto a plane orthogonal to the ray through $F_{\mathcal{A}3}$, but some license has been taken in distorting the figure to make the regions clearer. It can be shown that the fractional linear maps associated with trajectories in these networks take straight lines to straight lines,[21] and this applies also to projections onto a plane. The line through $S1$ and $S2$ is the separating boundary between the domains of

definition (also called returning cones) of the $\mathcal{A}$ and $\mathcal{B}$ cycles. We will denote this boundary by **b**. The dotted–dashed lines in Fig. 3 indicate the domains of definition of these two cycles as well as two others, $\mathcal{C}$ and $\mathcal{D}$, to be introduced below. The large letters show which cycle applies in which region. S_1 is the intersection of the stable manifold, $W^s_{\mathcal{B}}$, of $F_{\mathcal{B}3}$ and the separating boundary, **b** ($S_1 = W^s_{\mathcal{B}} \cap \mathbf{b}$). S_2 is the intersection of the unstable manifold, $W^u_{\mathcal{B}}$, of $F_{\mathcal{B}3}$ and **b** ($S_2 = W^u_{\mathcal{B}} \cap \mathbf{b}$). S_3 is the intersection of $W^u_{\mathcal{B}}$ and the inverse image under $M_{\mathcal{A}}$ of $W^s_{\mathcal{B}}$ [$S_3 = W^u_{\mathcal{B}} \cap M^{-1}_{\mathcal{A}}(W^s_{\mathcal{B}})$]. The triangle with vertices $F_{\mathcal{B}3}$, S_1, and S_3 is mapped into the two shaded regions (images of the $\mathcal{A}$ and $\mathcal{B}$ domain parts of the triangle under their respective mappings). Thus, this large triangle is invariant, as is the smaller triangle composed of the two shaded regions.

This picture changes as the bifurcation parameter, c, is changed. All of the key points and lines defining the regions shift. Furthermore, the domains of definition of each cycle also shift. These domains can be calculated from the cycle maps as can all of the manifolds and their images (for details, see Ref. 21). As shown in Fig. 3, when $c = 1.2546$, the vertex $F_{\mathcal{B}3}$ lies in the interior of the returning cone for $\mathcal{B}$ and the vertices S_1 and S_2 lie on the boundary, of course, but away from the vertices of the (projected) returning cones. S_3 lies in the interior of the returning cone for $\mathcal{A}$.

If we now increase the bifurcation parameter, c, when it reaches about 1.2703 the point S_1 leaves the returning cones for both cycles—in fact, it leaves the $(-,+,+)$ orthant altogether. Thus, the full trapping region is disrupted. However, since the shaded triangle of Fig. 3 is also a trapping region, and since S_1 is not part of it, this smaller trapping region persists. S_3 falls out of the returning cone for $\mathcal{A}$ at $c \approx 1.2762$, but S_3 is also not in the smaller trapping region. At $c \approx 1.2763$, $F_{\mathcal{B}3}$ falls out of the returning cone for $\mathcal{B}$, as does the other vertex of the shaded trapping region, $M_{\mathcal{A}}(S_3)$. At this point the smaller trapping region is also disrupted. However, the images of this trapping region under the mappings continue to shrink, and the attractor itself stays away from the boundary of the shaded region between $F_{\mathcal{B}3}$ and $M_{\mathcal{A}}(S_3)$. There is, in other words, a still smaller trapping region as yet unaffected by the fact that these points have left the returning cone for $\mathcal{B}$. This is demonstrated in the appendix.

The dynamical behavior changes significantly when $c \approx 1.2810$ at which point S_2 also leaves the returning cone for $\mathcal{B}$ (and $\mathcal{A}$), and moves onto the boundary between the returning cones for two other cycles, $\mathcal{C}$ and $\mathcal{D}$, respectively,

$$\mathcal{C}: 1011 \rightarrow \mathbf{1111} \rightarrow 1101 \rightarrow 1100 \rightarrow 1000 \rightarrow 0000 \rightarrow 0010 \rightarrow 0011 \rightarrow 1011,$$

$$\mathcal{D}: 1011 \rightarrow \mathbf{1001} \rightarrow 1101 \rightarrow 1100 \rightarrow 1000 \rightarrow 0000 \rightarrow 0010 \rightarrow 0011 \rightarrow 1011.$$

For c near this value, the attractor passes close to S_2 and so trajectories on the attractor fall outside the returning cones of $\mathcal{A}$ and $\mathcal{B}$ and into those of $\mathcal{C}$ and $\mathcal{D}$. Thus, the alphabet of the attractor increases to the set of four letters $\{\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}\}$.

TABLE III. Relative frequencies of occurrence of $\mathcal{A}$ and $\mathcal{B}$ in numerically generated sequences on trajectories of Eq. (5), for several values of the parameter c.

c	Pr($\mathcal{A}$)	Pr($\mathcal{B}$)
1.2546	0.68	0.32
1.2577	0.68	0.32
1.2678	0.63	0.37
1.2809	0.61	0.39

At $c \approx 1.2996$, the chaotic dynamics are lost altogether as a stable cycle appears, consisting of 56 transitions, represented by the sequence of simple cycles, $\mathcal{A}\mathcal{B}\mathcal{C}\mathcal{A}\mathcal{B}\mathcal{D}$. This cycle becomes stable when the fixed point of the return map lying on the dominant eigenvector of the map's matrix enters the returning cone for the cycle (see Ref. 21).

The symbolic representation allows us to introduce a novel type of bifurcation in chaotic dynamics—a change in parameter that leads to a new language. Although the alphabet does not change until $c \approx 1.2810$, the language does change. At the original value of the bifurcation parameter, $c = 1.2546$, the symbolic sequence $\mathcal{B}\mathcal{B}$ never occurs. This is a consequence of the fact that the attractor does not enter the part of the shaded trapping region in the $\mathcal{B}$ domain that is in the image of the $\mathcal{B}$ domain. However, when c reaches about 1.2577, the attractor moves into this region, and the symbolic sequence $\mathcal{B}\mathcal{B}$ begins to occur in trajectories on the attractor. Thus, the language is no longer $\mathcal{B}^k(\mathcal{A}^+\mathcal{B})^*$. After the bifurcation, the sequence $\mathcal{B}\mathcal{B}\mathcal{B}$ is still not possible as the accessible part of the $\mathcal{B}\mathcal{B}$ domain maps entirely into the $\mathcal{A}$ domain. The exact determination of this bifurcation value and the way in which $\mathcal{B}\mathcal{B}$ begins to appear in the attractor are explained in the Appendix.

The frequencies of occurrence of the letters may change continuously even between the language-bifurcation points. For example, the relative frequencies (approximate probabilities) of $\mathcal{A}$'s and $\mathcal{B}$'s in numerically generated sequences for a few values of c are listed in Table III.

Another way to look at these changes is in terms of the relative frequency of various lengths of strings of consecutive $\mathcal{A}$'s following a given $\mathcal{B}$. We list here the approximate probabilities of the string $\mathcal{B}\mathcal{B}$, and of 1,2 or 3 $\mathcal{A}$'s between successive $\mathcal{B}$'s for a few parameter values, based on numerical integrations (Table IV).

The probability of a $\mathcal{B}$ following a $\mathcal{B}$ increases with the parameter, and the probabilities of strings of $\mathcal{A}$'s generally decreases. As a result, the expected value of the length of a

TABLE IV. Relative frequencies of occurrence of $\mathcal{B}\mathcal{B}$ and $\mathcal{A}^1$, $\mathcal{A}^2$, and $\mathcal{A}^3$ between consecutive $\mathcal{B}$'s in trajectories of Eq. (5), for several values of the parameter c.

c	Pr($\mathcal{B}\mathcal{B}$)	Pr($\mathcal{A}^1$)	Pr($\mathcal{A}^2$)	Pr($\mathcal{A}^3$)
1.2546	0.00	0.51	0.21	0.13
1.2577	0.001	0.48	0.24	0.13
1.2678	0.12	0.46	0.24	0.09
1.2809	0.17	0.47	0.20	0.08

string of $\mathcal{A}$'s following a $\mathcal{B}$ decreases from about 2.1 when $c=1.2546$ to about 1.5 when $c=1.2809$.

The languages corresponding to the chaotic dynamics can be complicated, especially where the alphabet consists of four letters, and it can be rather difficult to determine the precise language for a given parameter value. Just before the bifurcation where chaos is lost, however, the language appears to be fairly simple. Long numerical integrations suggest that at $c=1.2995$, the language consists entirely of words made up of the subsequences $\mathcal{ABCABD}$ and $\mathcal{ABAABD}$. At $c=1.2986$ $\mathcal{ABD}$ may be repeated several times following an occurrence of $\mathcal{ABA}$ and for slightly smaller values of c the language appears to be even more complex.

V. THE INVERSE PROBLEM

A practical issue is to determine the network based solely on the observed dynamics. We call this the *inverse problem*. This means determining the inputs and the associated logical functions for each of the genes of the network. We believe that the issue of determining the quantitative values of parameters is a more difficult and less compelling problem than determining the underlying logical structure of the network. Given the current possibility of assaying the expression of thousands of genes simultaneously, it will be necessary to develop methods to determine functional interactions between genes based on the observed dynamics. Although the difficulty of carrying out this procedure in a realistic setting cannot be underestimated, several workers have tried to work out the genetic networks underlying specific systems.[23–27] Earlier proposals of general theoretical methods for carrying out the inverse problem were given by Glass and Young,[31] Liang and colleagues[32] and Akutsu and colleagues.[33,34] Here we illustrate how the inverse problem can be solved based on the example that shows chaotic dynamics that we discussed in Sec. III.

We assume that we know the following:

(1) No gene has self-input. This means that the logical function controlling a given gene does not depend on the logical state of that gene.
(2) The sequence of logical states observed during chaotic dynamics. This implies that we are able to measure the levels of the four variables over time and classify them into two levels, high (1) and low (0), by applying a threshold operation. Thus, we assume that we know the logical sequences associated with the two cyclic sequences $\mathcal{A},\mathcal{B}$.
(3) The sign of the rate of change of each of the variables for each of the states. By Eq. (5), the rates of change of each variable only depends on the orthant in phase space. Further, the sign of the rate of change of each variable is immediately determined from the truth table in Table II. For example, from the first line in Table II, we know that if all the variables are low, then variable w will be decreasing, whereas variables x,y,z will be increasing.

Thus, following a trajectory on the attractor, we can partially fill in the truth table as in Table V. The blanks in the truth table are associated with the five states that are not part of the chaotic attractor (bold lines in Fig. 1).

TABLE V. Partial truth table reconstructed from a trajectory on the chaotic attractor.

$(wxyz)_t$	$(wxyz)_{t+1}$
(0000)	0111
(0001)	
(0010)	1111
(0011)	1011
(0100)	
(0101)	
(0110)	0101
(0111)	0001
(1000)	0001
(1001)	0101
(1010)	
(1011)	1101
(1100)	1000
(1101)	1100
(1110)	
(1111)	0101

Since we have assumed that no gene has self-input, we can continue solving the inverse problem by assuming that each gene is a logical function of the other three, e.g., w depends on x,y,z. Consequently, we can rewrite the single truth table above, as four separate truth tables, one for each of the genes (Table VI). The blanks are associated with values that are not determined by the entries in the truth Table V, that is, edges on the N-cube with neither vertex on the attractor (see Fig. 1). Notice that only three values are not determined. Potentially, these entries could be filled in with either a 1 or 0, leading to a total of eight different networks that are consistent with the information given. However, if we further assume that each gene in the network is a function of only two of the other genes in the network [this is the rule governing the construction of Eq. (5)], then the information is adequate to reconstruct Table I. For example, consider the control of gene x. From the observation that states $wyz_t=010$ and $wyz_t=011$ are associated with different values of x_{t+1}, we know that x must be a function of z. Similarly, since $wyz_t=011$ and $wyz_t=111$ are associated with different values of x_{t+1}, we know that x must be a function of w. Thus, x is a function of w and z and the information is adequate to determine the complete truth table for x. In a similar fashion, we can determine that z is a function of xy, and we can reconstruct the truth tables in Table I. We emphasize that the

TABLE VI. Partial truth tables for each variable reconstructed from a trajectory on the chaotic attractor, assuming no self-input.

$(xyz)_t$	w_{t+1}	$(wyz)_t$	x_{t+1}	$(wxz)_t$	y_{t+1}	$(wxy)_t$	z_{t+1}
(000)	0	(000)	1	(000)	1	(000)	1
(001)	0	(001)		(001)	1	(001)	1
(010)	1	(010)	1	(010)	0	(010)	
(011)	1	(011)	0	(011)	0	(011)	1
(100)	1	(100)	0	(100)	0	(100)	1
(101)	1	(101)	1	(101)	0	(101)	1
(110)	0	(110)		(110)	0	(110)	0
(111)	0	(111)	1	(111)	0	(111)	1

ambiguity in three values before invoking the 2-input rule is a result of this particular network's attractor. Other nets with attractors that cover more of phase space may have no ambiguity even without the 2-input rule.

Although, this is an artificial example, it does give a method that can be used to reconstruct qualitative information about the interactions in a complex network based on limited information about the dynamics in the network.

VI. CONCLUSIONS

In this paper, we have shown how symbolic methods that arise in the study of computation theory can be applied to represent qualitative dynamics in models of gene networks. Though application of symbolic methods to nonlinear dynamics has a long history, the current work emphasizes the computational aspects of dynamical systems and may lead to novel ways to develop dynamical implementations of automata. The notion of associating bifurcations in dynamics in chaotic systems with changes in the associated language may be particularly useful.

The extent to which the methods here are applicable to real genetic systems is not known. The current equations are not realistic for many reasons: (i) control of gene expression is not on or off but is graded; (ii) there may be time delays associated with synthesis or degradation of gene products not accounted for here; (iii) decay rates of different gene products are different; (iv) although a single gene product might control expression of many different genes, the threshold levels for activation and/or inhibition may be different for different targets. However, all these represent quantitative changes in the equations, and the extent to which these changes influence the qualitative properties of the equations is still largely unknown. Further, it is intriguing that a recent analysis of gene expression in sea urchin concludes that the gene control can be approximated by a logical function based on the presence or absence of relevant factors that control the gene expression.[35]

In organisms, development and function usually appear to be orderly, and it is reasonable to question the relevance of chaotic dynamics to gene control mechanisms. The number of genes in humans is not known, but is likely of the order of 100 000 genes. The mechanisms of control of individual genes are now being worked out. It now appears likely that the expression of individual genes will be controlled by multiple inputs (e.g., the regulatory circuit worked out for a sea urchin gene had seven target sites for DNA binding proteins[35]). Theoretical models of high dimensional randomly constructed model gene networks (e.g., using the notation in Sec. II, with $N>50$ and $K>7$), showed that the usual circumstance is that the dynamics in such networks are chaotic.[36] Based on these observations, it might be reasonable to expect that gene networks in organisms could operate in chaotic regimes. However, the theoretical work assumes random networks with randomly constructed truth tables, and such assumptions probably do not hold in real organisms. It seems reasonable to assume that the truth tables for control of gene expression in organisms are not random. For example, Kauffman has hypothesized that regulation of gene expression in organisms is carried out by functions called ''canalizing'' in which one or more of the input variables forces the output to a fixed value.[13] The effects of incorporating canalizing functions in model gene control networks needs further analysis. Although, genetic networks may not operate in chaotic regimes, the current work stresses the analysis of these networks from a qualitative, computational perspective that may be represented symbolically. Finally, even if the current formalism cannot capture the qualitative aspects of real gene networks, it is intriguing that synthetic networks built out of genetic components[37,38] have qualitative dynamic properties that are well represented by the differential equations here.

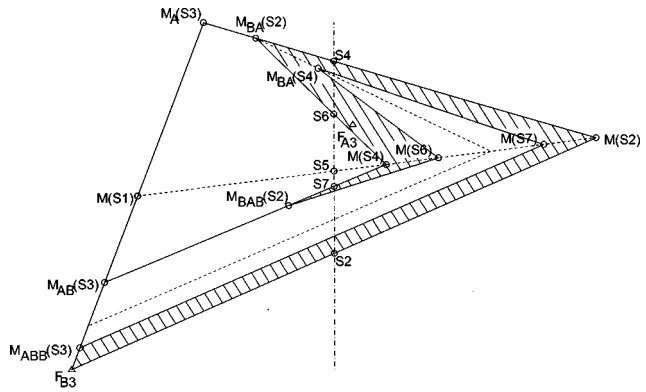

FIG. 4. Sketch of a smaller trapping region than the one in Fig. 3 ($c=1.2546$). Note that the triangular region in the $\mathcal{B}$ domain with vertex $M_{\mathcal{BAB}}(S_2)$ is also in the image of the $\mathcal{B}$ domain [the triangle with vertices $M_{\mathcal{B}}(S_1)$, $M_{\mathcal{B}}(S_2)$, and $F_{\mathcal{B}3}$] and therefore trajectories passing through this region contain symbolic subsequences $\mathcal{BB}$.

ACKNOWLEDGMENTS

We thank Thomas Mestl and Asa Ben-Hur for helpful conversations. This research has been supported by grants from NSERC and FCAR.

APPENDIX

We wish to locate the bifurcation point where the language associated with the dynamics on the attractor changes to allow the substring $\mathcal{BB}$, and to understand its appearance.

The trapping region discussed by Mestl *et al.*[20] (Fig. 3), contains a part of the intersection of the $\mathcal{B}$ domain and the image of the $\mathcal{B}$ domain [$M_{\mathcal{B}}$ maps the triangle with vertices S_1, S_2 and $F_{\mathcal{B}3}$ into the triangle with vertices $M(S_2)$, $M(S_1)$ and $F_{\mathcal{B}3}$]. Trajectories passing through this intersection contain the symbolic subsequence $\mathcal{BB}$, so we will call it the $\mathcal{BB}$ domain. But this trapping region is by no means minimal. Under iterations of the mappings, it maps into itself, of course, and these images shrink. With the original parameter value, $c=1.2546$, two more iterations from the shaded region of Fig. 3 lead to a trapping region (shaded region in Fig. 4) that still overlaps with the $\mathcal{BB}$ domain. The way in which these two mappings occur can be traced by means of the labeled points in the sketch. Every time a region crosses the separating boundary, **b**, between the $\mathcal{A}$ and $\mathcal{B}$ domains (the dotted–dashed line through S_2 and S_4) it folds on the next

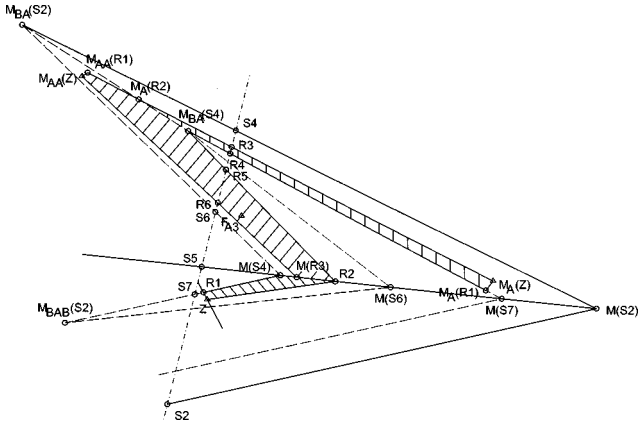

FIG. 5. Sketch of a still smaller trapping region that does not intersect the $\mathcal{BB}$ domain ($c=1.2546$).

iteration since part of it is subject to $M_{\mathcal{A}}$ and the other part to $M_{\mathcal{B}}$. The fold always first occurs on the image of the separating boundary, $M(\mathbf{b})$, the segment from $M(S_2)$ to $M(S_4)$.

Consider first the part of the lowest shaded strip in Fig. 4 that lies in the $\mathcal{B}$ domain (with lower edge from $F_{\mathcal{B}3}$ to S_2). Under iteration of $M_{\mathcal{B}}$, all points in this piece will eventually be mapped out into the $\mathcal{A}$ domain due to expansion along $W^u_{\mathcal{B}}$ (the lower edge), with the exception of points along the stable manifold of $F_{\mathcal{B}3}$ [the segment from $M_{\mathcal{ABB}}(S_3)$ to $F_{\mathcal{B}3}$]. No other part of the shaded region is mapped back into this piece, so we may clip it off and still have a trapping region. However, further iteration of this clipped trapping region never entirely avoids the $\mathcal{BB}$ domain. Note that the small triangle with tip at $M_{\mathcal{BAB}}(S_2)$ also extends into the $\mathcal{BB}$ domain. Points on the $\mathcal{B}$ side of this part of the trapping region follow the sequence of mappings $\mathcal{BAB}$ and under iteration of this combined mapping either come back into the same region or are ejected across **b** into the $\mathcal{A}$ domain. However, some points on the other side are sent back across **b** again under $M_{\mathcal{AAB}}$. To demonstrate that points in this region must be on transients and not on the attractor, we look more closely for a yet smaller trapping region.

Consider the shaded region shown in Fig. 5. The point marked with a triangle and labeled Z is an unstable fixed point of the composite map, $M_{\mathcal{AAB}}$. Thus, it and its images, $M_{\mathcal{A}}(Z)$ and $M_{\mathcal{AA}}(Z)$ (also marked by triangles) form a period-3 cycle of the full return map. R_1 is the point at which the stable manifold of the $\mathcal{AAB}$ map at Z intersects the line from $M(S_4)$ to $M_{\mathcal{BAB}}(S_2)$, as shown. R_2 is the point at which the unstable manifold of the $\mathcal{AAB}$ map at Z intersects the fold line $M(\mathbf{b})$. We note that although R_2 is not in the domain of definition (returning cone) for the map $M_{\mathcal{AAB}}$, it lies in the image of this domain. In fact, the part of the unstable manifold at Z that lies on the line segment from Z to R_2 but in the returning cone for $M_{\mathcal{AAB}}$ is mapped to the entire line segment from Z to R_2. R_3, R_4, R_5, and R_6 are the intersection points of the edges of two strips of the shaded region with the separating boundary, **b**. Note that $M(R_6)=R_2$ since R_6 lies on the image under $\mathcal{AA}$ of the unstable manifold of Z as well as on **b**, so under $M_{\mathcal{B}}$ it maps to the unstable manifold of Z and to $M(\mathbf{b})$.

The shaded region in Fig. 5 is a trapping region. This can be shown by considering it as the union of five polygonal pieces defined by the following sets of vertices:

T_1: $Z, R_1, M(S_4), R_2$,
T_{2a}: $M_{\mathcal{A}}(Z), M_{\mathcal{A}}(R_1), R_4, R_3$,
T_{2b}: $R_3, R_4, M_{\mathcal{BA}}(S_4), M_{\mathcal{A}}(R_2)$,
T_{3a}: $R_6, R_5, R_2, M(R_3)$,
T_{3b}: $M_{\mathcal{AA}}(Z), M_{\mathcal{AA}}(R_1), M_{\mathcal{A}}(R_2), M_{\mathcal{BA}}(S_4), R_5, R_6$.

Clearly, $M_{\mathcal{A}}(T_1)=T_{2a}\cup T_{2b}$.

Also, $M_{\mathcal{A}}(T_{2a})\subset T_{3a}\cup T_{3b}$, since $M_{\mathcal{A}}(Z), M_{\mathcal{A}}(R_1)$, and R_3 map to $M_{\mathcal{AA}}(Z), M_{\mathcal{AA}}(R_1)$, and $M(R_3)$, while R_4 lies on the line segment between R_3 and R_6 so that $M(R_4)$ lies on the line segment between $M(R_3)$ and $M(R_6)=R_2$. By a similar argument, $M(R_3)$ lies between $M(S_4)$ and R_2.

$M_{\mathcal{B}}(T_{2b})\subset T_1$. We have shown that $M(R_3)$ and $M(R_4)$ lie on the segment from $M(S_4)$ to R_2. A calculation is required to show that $M_{\mathcal{BA}}(S_4)$ and $M_{\mathcal{A}}(R_2)$ map under $M_{\mathcal{B}}$ into T_1 when $c=1.2546$.

$M_{\mathcal{A}}(T_{3a})\subset T_{3a}\cup T_{3b}$. Under $M_{\mathcal{A}}$, R_2 maps to $M_{\mathcal{A}}(R_2)$ and $M(R_3)$ maps to the line segment from $M_{\mathcal{BA}}(S_4)$ to $M_{\mathcal{A}}(R_2)$. R_6 maps to R_2 and R_5 maps to the line segment from $M(R_3)$ to R_2. Note that the unstable fixed point, $F_{\mathcal{A}3}$, for the map $M_{\mathcal{A}}$ lies in T_{3a}.

$M_{\mathcal{B}}(T_{3b})\subset T_1$. As shown above, R_5 and R_6 map to the segment from $M(S_4)$ to R_2. $M_{\mathcal{AA}}(Z)$ and $M_{\mathcal{AA}}(R_1)$ map under $M_{\mathcal{B}}$ to Z and R_1, respectively. And finally, as mentioned above, $M_{\mathcal{BA}}(S_4)$ and $M_{\mathcal{A}}(R_2)$ map under $M_{\mathcal{B}}$ into T_1 when $c=1.2546$.

The crucial point is that the $\mathcal{AAB}$ map has positive eigenvalues so that points on one side of the stable manifold at Z (the line through Z and R_1) stay on the same side under iteration of $M_{\mathcal{AAB}}$. Points in the returning cone of $\mathcal{AAB}$ on the T_1 side stay on this side under $M_{\mathcal{AAB}}$ and move away from the stable manifold, while points on the other side also move away from the stable manifold, eventually crossing into the $\mathcal{B}$ domain, and in particular, into the part of the $\mathcal{B}$ domain where $\mathcal{BB}$ occurs. Thus, it is essential to stay on the T_1 side of the stable manifold at Z. This can be ensured as long as $M_{\mathcal{AB}}(R_2)$ lies on the T_1 side [$M_{\mathcal{BAB}}(S_4)$ lies on the T_1 side if $M_{\mathcal{AB}}(R_2)$ does].

Intuitively, the reason why trajectories must eventually leave the part of the trapping region of Fig. 4 that lies in the $\mathcal{BB}$ domain is that even though they may return after $\mathcal{BAB}$ ejects them and $\mathcal{AAB}$ sends them back, every iteration of $\mathcal{BAB}$ maps part of this region across the stable manifold of $\mathcal{AAB}$ at Z, so that eventually all trajectories escape and enter the trapping region of Fig. 5 (except a set of measure zero, the part of the stable manifold of the unstable fixed point of $\mathcal{BAB}$ that lies in the region).

Now we are ready to consider changing c again. First, if we *decrease* c to ≈ 1.2537 then $M_{\mathcal{BAB}}(S_2)$ lies in the $\mathcal{A}$ domain, while the rest of the picture remains qualitatively unchanged so there is no possibility of $\mathcal{BB}$ occurring, even based on the trapping region of Fig. 4. For larger values of c we need the trapping region of Fig. 5, and this remains a trapping region until $c\approx 1.2577$ at which point $M_{\mathcal{AB}}(R_2)$ crosses over the stable manifold of Z and leaves T_1. It is not easy to prove that this is sufficient to ensure that $\mathcal{BB}$ occurs on the attractor [it would require showing that trajectories

pass arbitrarily close to $M_{AB}(R_2)$ or, equivalently, R_2 infinitely often], but numerical integrations suggest that this is the case.

[1] S. A. Fodor, ''Massively parallel genomics,'' Science **277**, 393–395 (1997).
[2] R. Sapolsky, L. Hsie, A. Berno, G. Ghandour, M. Mittmann, and J. B. Fan, ''High-throughput polymorphism screening and genotyping with high-density oligonucleotide arrays,'' Biomol. Eng. **14**, 187–192 (1999).
[3] S. Wiggins, *Introduction to Applied Nonlinear Dynamical Systems and Chaos* (Springer-Verlag, New York, 1990).
[4] D. Lind and B. Marcus, *An Introduction to Symbolic Dynamics and Coding* (Cambridge University Press, Cambridge, 1995).
[5] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation* (Addison-Wesley, Reading, MA, 1979).
[6] M. Sipser, *Theory of Computation* (PWS, Boston, 1997).
[7] L. Blum, F. Cucker, M. Shub, and S. Smale, *Complexity and Real Computation* (Springer-Verlag, New York, 1997).
[8] W. S. McCulloch and W. Pitts, ''A logical calculus of the ideas immanent in nervous activity,'' Bull. Math. Biophys. **5**, 115–133 (1943).
[9] S. A. Kauffman, ''Metabolic stability and epigenesis in randomly constructed genetic networks,'' J. Theor. Biol. **22**, 437–467 (1969).
[10] J. J. Hopfield, ''Neurons with graded responses have collective computational properties like those of two-state-neurons,'' Proc. Natl. Acad. Sci. U.S.A. **81**, 3088–3092 (1984).
[11] H. T. Siegelmann, *Neural Networks and Analog Computation: Beyond the Turing Limit* (Birkhauser, Boston, 1999).
[12] R. Thomas, ''Boolean formalization of genetic control circuits,'' J. Theor. Biol. **42**, 563–585 (1973); R. Thomas and R. D'Ari, *Biological Feedback* (Chemical Rubber Company, Boca Raton, FL, 1990).
[13] S. A. Kauffman, *Origins of Order: Self-Organization and Selection in Evolution* (Oxford University Press, Oxford, 1993).
[14] L. Glass and S. A. Kauffman, ''Cooperative components, spatial localization and oscillatory cellular dynamics,'' J. Theor. Biol. **34**, 219–237 (1972).
[15] L. Glass and S. A. Kauffman, ''The logical analysis of continuous, nonlinear biochemical control networks,'' J. Theor. Biol. **39**, 103–129 (1973).
[16] L. Glass, ''Combinatorial and topological methods in nonlinear chemical kinetics,'' J. Chem. Phys. **63**, 1325–1335 (1975).
[17] L. Glass and J. S. Pasternack, ''Prediction of limit cycles in mathematical models of biological oscillations,'' Bull. Math. Biol. **40**, 27–44 (1978).
[18] L. Glass and J. S. Pasternack, ''Stable oscillations in mathematical models of biological control systems,'' J. Math. Biol. **6**, 207–223 (1978).
[19] T. Mestl, E. Plahte, and S. W. Omholt, ''Periodic solutions in systems of piecewise-linear differential equations,'' Dyn. Stab. Syst. **10**, 179–193 (1995).
[20] T. Mestl, C. Lemay, and L. Glass, ''Chaos in high dimensional neural and gene networks,'' Physica D **98**, 33–52 (1996).
[21] R. Edwards, ''Analysis of continuous-time switching networks,'' Physica D **146**, 165–199 (2000).
[22] R. Edwards and L. Glass, ''Combinatorial explosion in model gene networks,'' Chaos **10**, 691–704 (2000).
[23] H. H. McAdams and L. Shapiro, ''Circuit simulation of genetic networks,'' Science **269**, 650–656 (1995).
[24] D. Thieffry and R. Thomas, ''Dynamical behavior of biological regulatory networks, II. Immunity control in bacteriophage lambda,'' Bull. Math. Biol. **57**, 277–295 (1995).
[25] J. Reinitz and D. H. Sharp, ''Mechanism of *eve* stripe formation,'' Mech. Dev. **49**, 133–158 (1995).
[26] G. Marnellos and E. Mjolsness, ''A gene network approach to modeling early neurogenesis in drosophila,'' in *Pacific Symposium on Biocomputing '98*, edited by R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein (World Scientific, Singapore, 1998), pp. 30–41.
[27] L. Mendoza and E. Alvarez-Buylla, ''Dynamics of the genetic regulatory network for *Arabidopsis thaliana* flower morphogenesis,'' J. Theor. Biol. **193**, 307–319 (1998).
[28] B. J. Meyer and M. Ptashne, ''Gene regulation at the right operator (O_R) of bacteriophage λ. III. λ repressor directly activates gene transcription,'' J. Mol. Biol. **139**, 195–205 (1980).
[29] J. P. Crutchfield, ''The calculi of emergence: Computations, dynamics, and induction,'' Physica D **75**, 11–54 (1994).
[30] C. Moore and P. Lakdawala, ''Queues, stacks, and transcendentality at the transition to chaos,'' Physica D **135**, 24–40 (2000).
[31] L. Glass and R. Young, ''Structure and dynamics of neural network oscillators'' Brain Res. **179**, 207–218 (1979); D. T. Kaplan and L. Glass, *Understanding Nonlinear Dynamics* (Springer-Verlag, New York, 1995), pp. 70–73.
[32] S. Liang, S. Fuhrman, and R. Somogyi. ''REVEAL, A general reverse engineering algorithm for inference of genetic network architectures,'' in *Pacific Symposium on Biocomputing '98*, edited by R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein (World Scientific, Singapore, 1998), vol. 3, pp. 18–29.
[33] T. Akutsu, S. Miyano, and S. Kuhara, ''Identification of genetic networks from a small number of gene expression patterns under the Boolean network model,'' in *Pacific Symposium on Biocomputing '99*, edited by R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein (World Scientific, Singapore, 1999), vol. 4, pp. 17–24.
[34] T. Akutsu, S. Miyano, and S. Kuhara, ''Algorithms for inferring qualitative models of biological networks,'' in *Pacific Symposium on Biocomputing 2000*, edited by R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein (World Scientific, Singapore, 2000), vol. 5, pp. 290–301.
[35] C.-H. Yuh, H. Bolouri, and E. H. Davidson, ''Genomic cis-regulatory logic: Experimental and computational analysis of a sea urchin gene,'' Science **279**, 1896–1902 (1998).
[36] T. Mestl, R. J. Bagley, and L. Glass, ''Common chaos in arbitrarily complex feedback networks,'' Phys. Rev. Lett. **79**, 653–656 (1997).
[37] M. B. Elowitz and S. Leibler, ''A synthetic oscillatory network of transcriptional regulators,'' Nature (London) **403**, 335–338 (2000).
[38] T. S. Gardner, C. R. Cantor, and J. J. Collins, ''Construction of a genetic toggle switch in *Escherichia coli*,'' Nature (London) **403**, 339–342 (2000).