Hindawi Publishing Corporation
EURASIP Journal on Bioinformatics and Systems Biology
Volume 2007, Article ID 97356, 8 pages
doi:10.1155/2007/97356

Research Article

Fixed Points in Discrete Models for Regulatory Genetic Networks

Dorothy Bollman,[1] Omar Colón-Reyes,[1] and Edusmildo Orozco[2]

[1] *Departament of Mathematical Sciences, University of Puerto Rico, Mayaguez, PR 00681, USA*
[2] *Department of Computer Science, University of Puerto Rico, Río Piedras, San Juan, PR 00931-3355, USA*

Received 1 July 2006; Revised 22 November 2006; Accepted 20 February 2007

Recommended by Tatsuya Akutsu

It is desirable to have efficient mathematical methods to extract information about regulatory iterations between genes from repeated measurements of gene transcript concentrations. One piece of information is of interest when the dynamics reaches a steady state. In this paper we develop tools that enable the detection of steady states that are modeled by fixed points in discrete finite dynamical systems. We discuss two algebraic models, a univariate model and a multivariate model. We show that these two models are equivalent and that one can be converted to the other by means of a discrete Fourier transform. We give a new, more general definition of a linear finite dynamical system and we give a necessary and sufficient condition for such a system to be a fixed point system, that is, all cycles are of length one. We show how this result for generalized linear systems can be used to determine when certain nonlinear systems (monomial dynamical systems over finite fields) are fixed point systems. We also show how it is possible to determine in polynomial time when an ordinary linear system (defined over a finite field) is a fixed point system. We conclude with a necessary condition for a univariate finite dynamical system to be a fixed point system.

1. INTRODUCTION

Finite dynamical systems are dynamical systems on finite sets. Examples include cellular automata and Boolean networks, (e.g., [1]) with applications in many areas of science and engineering (e.g., [2, 3]), and more recently in computational biology (e.g., [4–6]). A common question in all of these applications is how to analyze the dynamics of the models without enumerating all state transitions. This paper presents partial solutions to this problem.

Because of technological advances such as DNA microarrays, it is possible to measure gene transcripts from a large number of genes. It is desirable to have efficient mathematical methods to extract information about regulatory iterations between genes from repeated measurements of gene transcript concentrations.

One piece of information about regulatory iterations of interest is when the dynamics reaches a steady state. In the words of Fuller (see [7]): "this paradigm closely parallels the goal of professionals who aim to understand the flow of molecular events during the progression of an illness and to predict how the disease will develop and how the patient will respond to certain therapies."

The work of Fuller et al. [7] serves as an example. When the gene expression profile of human brain tumors was analyzed, these were divided into three classes—high grade, medium grade, and low grade. A key gene expression event was identified, which was a high expression of insulin-like growth factor binding protein 2 (IGFBP2) occurring only in high-grade brain tumors. It can be assumed that gene expression events were initiated at some stages in low-level tumors and may have led to the state when IGFBP2 is activated. The activation of IGFBP2 can be understood to be a steady state. If we model the kinetics and construct a model that reconstructs the genetic regulatory network that activates during the brain tumor process, then we may be able to predict the convergence of events that lead to the activation of IGFBP2. In the same way, we also want to know what happens in the next step following the activation of IGFBP2. Our goal is to develop tools that will enable this type of analysis in the case of modeling gene regulatory networks by means of discrete dynamical systems.

The use of polynomial dynamical systems to model biological phenomena, in particular gene regulatory networks, have proved to be as valid as continuous models. Laubenbacher and Stigler (see [6]) point out, for example, that most ordinary differential equations models cannot be solved analytically and that numerical solutions of such time-continuous systems necessitate approximations by time-discrete systems, so that ultimately, the two types of models are not that different.

Once a gene regulatory network is modeled, in our case by finite fields, or by finitely generated modules, we obtain a finite dynamical system. Our goal is to determine if the dynamical system represents a steady-state gene regulatory network (i.e., if every state eventually enters a steady state). This is a crucial task. Shmulevich et al. (see [8]) have shown that the steady-state distribution is necessary in order to compute the long term influence that is a measure of gene impact over other genes.

The rest of the paper is organized as follows. In Section 2 we give some basic definitions and facts about finite dynamical systems and their associated state spaces. In Section 3 we discuss multivariate and univariate finite field models for genetic networks and show that they are equivalent. Each of the models can be converted to the other by a discrete Fourier transform. Section 4 is devoted to fixed point systems. We give a new definition of linear finite dynamical systems and give necessary and sufficient conditions for such a system to be a fixed point system. We review results concerning monomial fixed point systems and show how our results concerning linear systems can be used to determine when a monomial finite dynamical system over an arbitrary finite field is a fixed point system. We show how fixed points can be determined in the univariable model by solving a polynomial equation over a finite field and we give a necessary condition for a finite dynamical system to be a fixed point system. Finally, in Section 5 we discuss some implementation issues.

2. PRELIMINARIES

A *finite dynamical system* (fds) is an ordered pair (X, f) where X is a finite set and f is a function that maps X into itself, that is, $f : X \to X$. The *state space* of an fds (X, f) is a digraph (i.e., directed graph) whose nodes are labeled by the elements of X and whose edges consist of all ordered pairs $(x, y) \in X \times X$ such that $f(x) = y$. We say that two finite dynamical systems are *isomorphic* if there exists a graph isomorphism between their state spaces.

Let $G = (V, E)$ be a digraph. A path in G of the form $(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, v_1)$, where $v_1, v_2, \ldots, v_n$ are distinct members of V is a *cycle* of *length* n. We define a *tree* to be a digraph $T = (V, E)$ which has a unique node v_0, called the *root* of T, such that (a) $(v_0, v_0) \in E$, (b) for any node $v \neq v_0$, there is a path from v to v_0, (c) T has no "semicycles" (i.e., alternate sequence of nodes and edges $v_1, x_1, v_2, \ldots, x_n, v_{n+1}$, $n \neq 0$, where $v_1 = v_{n+1}$ and each x_i is $(v_i, v_{i+1}$ or $v_{i+1}, v_i))$ other than the trivial one $(v_0, (v_0, v_0), v_0)$. (Such a tree with the edge (v_0, v_0) deleted is sometimes called an "in-tree" with "sink" v_0.)

Let T be a tree, let $nT = \bigcup_{i=1}^{n} T_i$ be the union of n copies $T_1, T_2, \ldots, T_n$ of T, and let r_i be the root of T_i. Define $T^{(n)}$ to be the digraph obtained from nT by deleting the edges (r_i, r_i), $i = 1, 2, \ldots, n$, and adjoining the edges (r_i, r_j), $i, j = 1, 2, \ldots, n$, where $j = i + 1 \bmod n$. We call $T^{(n)}$ an *n-cycled tree*. Note that by definition, every tree is an n-cycled tree with $n = 1$. Note also that by definition, a digraph $T_r = (\{r\}, \{r, r\})$ consisting of a single trivial cycle is a tree and hence every cycle of length n is isomorphic to nT_r and hence is an n-cycled tree.

The *product* of two digraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, denoted $G_1 \times G_2$, is the digraph $G = (V, E)$ where $V = V_1 \times V_2$ (the Cartesian product of V_1 by V_2) and $E = \{((x_1, y_1), (x_2, y_2)) \in V \times V : (x_1, x_2) \in E_1$ and $(y_1, y_2) \in E_2\}$. The following facts follow easily from the definitions. Lemmas 1, 3, and 4 have been noted in [9].

Lemma 1. *The state space of an fds is the disjoint union of cycled trees.*

Of special interest are those fds whose state space consists entirely of trees. Such an fds is called a *fixed point system* (fps).

For any finite set X we call $f : X \to X$ *nilpotent* if there exists a unique $x_0 \in X$ such that $f^k(X) = x_0$ for some positive integer k.

Lemma 2. *The state space of an fds (X, f) is a tree if and only if f is nilpotent. Hence (X, f) is an fps if f is nilpotent.*

Proof. Suppose that the state space (X, f) is a tree with root x_0 and height k. Then $f^k(x) = x_0$ for all $x \in X$ and x_0 is the only node with this property. Hence f is nilpotent. Conversely, if f is nilpotent and $f^k(X) = x_0$, then by Lemma 1, the state space consists of an n-cycled tree and since x_0 is unique, $n = 1$. □

Example 1. Consider the fds (F_2^3, f), where $f : F_2^3 \to F_2^3$ is defined by $f(x, y, z) = (y, 0, x)$ and F_2 is the binary field. In this case f is a nilpotent function. The state space of (F_2^3, f) is a tree whose state space is shown in Figure 1.

Lemma 3. *The state space of an fds (X, f) is the union of cycles if and only if f is one-to-one.*

Lemma 4. *The product of a tree and a cycle of length l is a cycled tree whose cycle has length l.*

3. FINITE FIELD MODELS

A finite dynamical system constitutes a very natural discrete model for regulatory processes (see [10]), in particular genetic networks. Experimental data can be discretized into a finite set X of expression levels. A network consisting of n genes is then represented by an fds (X^n, f). The dynamics of the network is described by a discrete time series

$$f(s_0) = s_1, f(s_1) = s_2, \ldots, f(s_{k-2}) = s_{k-1}. \tag{1}$$

Special cases of the finite dynamical model are the Boolean model and finite field models. In the Boolean model,

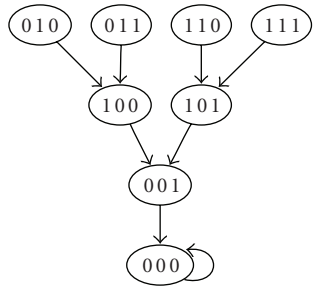

FIGURE 1: State space of (F_2^3, f), where $f(x, y, z) = (y, 0, x)$ over F_2.

either a gene can affect another gene or not. In a finite field model, one is able to capture graded differences in gene expression. A finite field model can be considered as a generalization of the Boolean model since each Boolean operation can be expressed in terms of the sum and product in Z_2. In particular,

$$\begin{aligned} x \wedge y &= x \cdot y, \\ x \vee y &= x + y + x \cdot y, \\ \widetilde{x} &= x + 1. \end{aligned} \tag{2}$$

Two types of finite field models have emerged, the multivariate model [6] and the univariable model [11]. The multivariate model is given by the fds (F_q^n, f), where F_q^n represents the set of n-tuples over the finite field F_q with q elements. Each coordinate function f_i gives the next state of gene i, given the states of the other genes. The univariate model is given by the fds (F_{q^n}, f). In this case, each value of f represents the next states of the n genes, given the present states.

The two types of finite field models can be considered equivalent in the following sense.

Definition 1. An fds (X, f) is *equivalent* to an fds (Y, g) if there is an epimorphism $\phi : X \to Y$ such that $\phi \circ f = g \circ \phi$.

It is easy to see that if two fds's are equivalent, then their state spaces are the same up to isomorphism. We can show that for any n-dimensional dynamical system (F_q^n, f) there is an equivalent one-dimensional system (F_{q^n}, g). To see this, consider a primitive element α of F_{q^n}, that is, a generator of the multiplicative group of $F_{q^n} - \{0\}$. Then there is a natural correspondence between F_q^n and F_{q^n}, given by

$$\phi_\alpha(x_0, \ldots, x_{n-1}) = x_0 + x_1\alpha + x_2\alpha^2 + \cdots + x_{n-1}\alpha^{n-1}. \tag{3}$$

Since for each $a \in F_{q^n}$ there exists unique $y_i \in F_q$ such that $a = y_0 + y_1\alpha + y_2\alpha^2 + \cdots + y_{n-1}\alpha^{n-1}$ we can define $g : F_{q^n} \to F_{q^n}$ as $g(a) = (\phi_\alpha \circ f)(y_0, \ldots, y_{n-1})$. Notice then that $g \circ \phi_\alpha = \phi_\alpha \circ f$ and therefore the dynamical systems g and f are equivalent.

One important consideration in choosing an appropriate finite field model for a genetic network is the complexity of the needed computational tasks. For example, the evaluation of a polynomial in n variables over F_q, q prime, can be done with $O(q^n/n)$ operations (see [12]) and hence, evaluating f in all n of its coordinates is $O(q^n)$, the same number of operations needed for the evaluation of a univariate polynomial over F_{q^n}. However, the complexity of the comparison of two values in F_q^n is $O(n)$, whereas the complexity of the comparison of two values in F_{q^n}, represented as described below, is $O(1)$.

Arithmetic in F_q^n, q prime, is integer arithmetic modulo q. Arithmetic in F_{q^n} is efficiently performed by table lookup methods, as shown below. Nonzero elements of F_{q^n} are represented by powers of a primitive element α. Multiplication is then performed by adding exponents modulo $q^n - 1$. For addition we make use of a precomputed table of values defined as follows. Every nonozero element of F_{q^n} has a unique representation in the form $1 + \alpha^i$ and the unique number $z(i)$, $0 \le z(i) \le q^n - 2$, such that $1 + \alpha^i = \alpha^{z(i)}$ is called the *Zech log* of i. Note that for $a \le b$, $\alpha^a + \alpha^b = \alpha^a(1 + \alpha^{b-a}) = \alpha^{a+z(b-a) \bmod q^n-1}$. Addition is thus performed by adding one exponent to the Zech log of the difference, which is found in a precomputed table. In order to construct a table of Zech logs for F_{q^n}, we first need a primitive polynomial, which can be found in any one of various tables (e.g., [13]).

Example 2. Let us construct a table of Zech logs for F_{32} using the primitive polynomial x^5+x^2+1. Thus, we have $\alpha^5 = \alpha^2+1$, where α is a root of $x^5 + x^2 + 1$. Continuing to compute the powers and making use of this fact, we have $\alpha^6 = \alpha^3 + \alpha$, $\alpha^7 = \alpha^4 + \alpha^2$, $\alpha^8 = \alpha^5 + \alpha^3 = \alpha^3 + \alpha^2 + 1, \ldots$, $\alpha^{31} = 1$. Now use these results to compute for each $i = 1, \ldots, 30$, the number $z(i)$ such that $\alpha^i + 1 = \alpha^{z(i)}$. For example, since $\alpha^5 = \alpha^2 + 1$, we have $\alpha^5 + 1 = \alpha^2$ and so $z(5) = 2$, and so forth. See Table 1.

Usually it is most convenient to choose the most appropriate model at the outset. However, at the cost of computing all possible values of the map, it is possible to convert one model to the other. The rest of this section is devoted to developing such an algorithm.

Definition 2. Let F be a field and let $\alpha \in F$ be an element of order d, that is, $\alpha^d = 1$ and no smaller power of α equals 1. The discrete Fourier transform (DFT) of blocklength d over F is defined by the matrix $T = [\alpha^{ij}], i, j = 0, 1, \ldots, d - 1$. The inverse discrete Fourier transform is given by $T^{-1} = d^{-1}[\alpha^{-ij}], i, j = 0, 1, \ldots, d - 1$, where d^{-1} denotes the inverse of the field element $d = 1 + 1 + \cdots + 1$ (d times).

It is easy to show that $TT^{-1} = I_d$, where I_d denotes the $d \times d$ identity matrix (see, e.g., [14]). Now an element in F_q, is of order d if and only if d divides $q - 1$. Thus, for every finite field F_q there is a DFT over F_q with block length $q - 1$ which is defined by $[\alpha^{ij}]$, $i, j = 0, 1, \ldots, q - 2$, where α is a primitive element of F_q. We denote such a DFT by $T_{q,\alpha}$.

Theorem 1. *Let* $B_0 = (\phi_\alpha \circ f)(0, \ldots, 0)$ *and for each* $i = 1, 2, \ldots, q^n - 1$, *let* $B_i = (\phi_\alpha \circ f)(a_{0,i}, a_{1,i}, \ldots, a_{n-1,i})$ *where*

Table 1: Zech Logs for F_{32}.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$z(i)$	18	5	29	10	2	27	22	20	16	4	19	23	14	13	24

i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
$z(i)$	9	30	1	11	8	25	7	12	15	21	28	6	26	3	17

α is a primitive element of F_{q^n} and where $a_{n-1,i}\alpha^{n-1} + \cdots + a_{1,i}\alpha + \alpha_{0,i} = \alpha^{i-1}$. Then g is given by the polynomial

$$A_{q^n-1}x^{q^n-1} + A_{q^n-2}x^{q^n-2} + \cdots + A_1x + A_0, \tag{4}$$

where $A_0 = B_0$ and

$$\begin{pmatrix} A_{q^n-1} \\ A_{q^n-2} \\ \vdots \\ A_1 \end{pmatrix} = -T_{q^n,\alpha} \begin{pmatrix} B_1 - A_0 \\ B_2 - A_0 \\ \vdots \\ B_{q^n-1} - A_0 \end{pmatrix}. \tag{5}$$

Proof. For each $i = 0, 1, \ldots, q^n - 2$, we have

$$\begin{aligned} B_{i+1} &= \phi_\alpha(f(a_{0,i+1}, a_{1,i+1}, \ldots, a_{n-1,i+1})) \\ &= g(\phi_\alpha(a_{0,i+1}, a_{1,i+1}, \ldots, a_{n-1,i+1})) \\ &= g(a_{0,i+1} + a_{1,i+1}\alpha + \cdots + a_{n-1,i+1}\alpha^{n-1}) = g(\alpha^i). \end{aligned} \tag{6}$$

Now every function defined on a finite field F_{q^n} can be expressed as a polynomial of degree not more than $q^n - 1$. Hence g is of the form (4) and it remains to show that the A_i are given by (5). For this we need only to solve the following system of equations:

$$\begin{aligned} g(\alpha^i) =& A_{q^n-1}(\alpha^i)^{q^n-1} + A_{q^n-2}(\alpha^i)^{q^n-2} \\ &+ \cdots + A_1\alpha^i + A_0, \quad i = 0, 1, \ldots, q^n - 2. \end{aligned} \tag{7}$$

Since α is a primitive element of F_{q^n}, we have $(\alpha^i)^{q^n-1} = 1$ and so

$$\begin{aligned} B_{i+1} - A_0 = g(\alpha^i) = & A_{q^n-1}(\alpha^i)^{q^n-1} + A_{q^n-2}(\alpha^i)^{q^n-2} \\ &+ \cdots + A_1\alpha^i, \quad i = 0, 1, \ldots, q^n - 2. \end{aligned} \tag{8}$$

Thus,

$$\begin{pmatrix} B_1 - A_0 \\ B_2 - A_0 \\ \vdots \\ B_{q^n-1} - A_0 \end{pmatrix} = d^{-1}T^{-1}_{q^n,\alpha} \begin{pmatrix} A_{q^n-1} \\ A_{q^n-2} \\ \vdots \\ A_1 \end{pmatrix}, \tag{9}$$

where $d^{-1} = (q^n - 1)^{-1} = -1$. The theorem then follows by applying $T_{q^n,\alpha}$ to both sides of this last equation. □

We illustrate the algorithm given by Theorem 1 with an example.

Example 3. A recent application involves the study and creation of a model for *lac* operon [15]. When the bacteria *E. Coli* is in an environment with lactose, then the lac operon turns on the enzymes that are needed in order to degrade lactose. These enzymes are *beta-galactisidase*, *Lactose Permease,* and *Thiogalactoside transectylase*. In [15], a continuous model is proposed that measures the rate of change in the concentration of these enzymes as well as the concentration of mRNA and intracellular lactose. In [16, 17], Laubenbacher and Stigler provide a discrete model for the lac operon given by

$$\begin{aligned} &(F_2^5, f(x_1, x_2, x_3, x_4, x_5)) \\ &\quad = (x_3, x_1, x_3 + x_2x_4 + x_2x_3x_4, (1 + x_2)x_4 \\ &\qquad + x_5 + (1 + x_2)x_4x_5, x_1), \end{aligned} \tag{10}$$

where x_1 represents mRNA, x_2 represents beta-galactosidase, x_3 represents allolactose, x_4 represents lactose, and x_5 represents permease. In order to find an equivalent univariate fds (f_{2^5}, g) we first find a primitive element α in F_{2^5}. This can be done by finding a "primitive polynomial," that is, an irreducible polynomial of degree 5 over F_2 that has a zero α in F_{2^5} that generates the multiplicative cyclic group of $F_{2^5} - \{0\}$. Such an α can be found either by trial and error or by the use of tables (see, e.g., [13]).

In our case, we choose α to be a zero of $x^5 + x^2 + 2$. Next, we compute B_i, $i = 0, 1, \ldots, 31$. By definition $B_0 = \phi_\alpha(f(0, 0, 0, 0, 0)) = \phi_\alpha(0, 0, 0, 0, 0) = 0$ and $B_i = \phi_\alpha(f(a_{0,i}, a_{1,i}, a_{2,i}, a_{3,i}, a_{4,i}))$ where $\alpha^{i-1} = a_{0,i} + a_{1,i}\alpha + a_{2,i}\alpha^2 + a_{3,i}\alpha^3 + a_{4,i}\alpha^4$ for $i = 1, 2, \ldots, 31$.

So, for example,

$$\begin{aligned} B_1 &= \phi_\alpha(f(1, 0, 0, 0, 0)) = \phi_\alpha(0, 1, 0, 0, 1) \\ &= \alpha + \alpha^4 = \alpha^{1+z(3)} = \alpha^{30}, \\ B_2 &= \phi_\alpha(f(0, 1, 0, 0, 0)) = \phi_\alpha(0, 0, 0, 0, 0) = 0. \end{aligned} \tag{11}$$

Continuing we find that $[B_1, B_2, \ldots, B_{31}] = [\alpha^{30}, 0, \alpha^5, \alpha^3, \alpha^3, \alpha^{26}, \alpha^2, \alpha^8, \alpha^{15}, \alpha^{20}, \alpha^9, \alpha^{26}, \alpha^5, \alpha^8, \alpha^{15}, \alpha^{15}, \alpha^{24}, \alpha^9, \alpha^{30}, \alpha^5, \alpha^8, \alpha^3, \alpha^{15}, \alpha^{26}, \alpha^8, \alpha^9, \alpha^{15}, \alpha^{28}, \alpha^8, \alpha^9, \alpha^3]$.

Multiplying by the 31×31 matrix $T_{32,\alpha} = [\alpha^{ij}]$, $0 \le i, j \le 30$, we obtain $[A_{31}, A_{30}, \ldots, A_1]$ and hence the equivalent univariate polynomial, which is $g(x) = x + \alpha^{22}x^2 + \alpha^2x^3 + \alpha^{11}x^4 + \alpha^{20}x^5 + x^6 + \alpha^{12}x^7 + \alpha^{25}x^8 + \alpha^{20}x^9 + \alpha^{20}x^{10} + \alpha^2x^{11} + \alpha^5x^{12} + \alpha^5x^{13} + \alpha^{23}x^{14} + \alpha^7x^{16} + \alpha^{27}x^{17} + \alpha^{20}x^{18} + \alpha^6x^{19} + \alpha^{20}x^{20} + \alpha^{27}x^{21} + x^{22} + \alpha^{27}x^{24} + x^{25} + \alpha^{27}x^{26} + \alpha^{16}x^{28}$.

As previously mentioned, the complexity of evaluating a polynomial in n variables over a finite field F_q is $O(q^n/n)$. The complexity of evaluating f in all of its n coordinates is thus $O(q^n)$ and the complexity of evaluating f in all points of F_{q^n} is thus $O(q^{2n})$. The computation of the matrix-vector product in (5) involves $O(q^{2n})$ operations over the field F_{q^n}. However, using any one of the number of classical fast algorithms, such as Cooley-Tukey (see, e.g., [14]), the number of operations can be reduced to $O(q^n n)$.

4. FIXED POINT SYSTEMS

A fixed point system (fps) is defined to be an fds whose state space consists of trees, that is, contains no cycles other than

trivial ones (of length one). The fixed point system problem is the problem of determining of an fds whether or not it is an fps. Of course one such method would be the brute force method whereby we examine sequences determined by successive applications of the state map to determine if any such sequence contains a cycle of length greater than one. The worst case occurs when the state space consists of one cycle. Consider such a multivariate fds (F_q^n, f). In order to recognize a maximal cycle, $f(a_1, a_2, \ldots, a_n), f^2(a_1, a_2, \ldots, a_n), \ldots,$ such an approach would require backtracking at each step in order to compare the most recent value $f^i(a_1, a_2, \ldots, a_n)$ with all previous values. An evaluation requires $O(q^n)$ operations, there are q^n such evaluations, and a comparison of two values requires n steps. The complexity of the complete process is thus $O(q^{2n} + n^2) = O(q^{2n})$.

To date, all results concerning the fixed point system problem are characterized in terms of multivariate fds. A solution to the fixed point system problem consists of characterizing such an fds (F_q^n, f) in terms of the structure of f. Ideally, such conditions should be amenable to implementations in polynomial time in n. In a recent work, Just [18] claims that if the class of regulatory functions contains the quadratic monotone functions $x_i \vee x_j$ and $x_i \wedge x_j$, then the fixed point problem for Boolean dynamical systems is NP-hard. In view of this result, it is unlikely that we can achieve the goal of a polynomial time solution to the fixed point problem, at least in the general case. However, the question arises if the above $O(q^{2n})$ result can be improved (to say $O(q^n)$) and also what are special cases of the fixed point problem that have polynomial time solutions.

In this section we give a polynomial solution to the special case of the fixed point problem for linear finite dynamical systems, we review known results for the nonlinear case, and we point out how our results concerning the general linear case for fds over finitely generated modules give a more complete solution to the case of monomial finite field dynamical systems over arbitrary finite fields. We conclude by proposing a new approach to the problem via univariate systems, we give an algorithm for determining the fixed points of a univariate system, and we give a necessary condition for a univariate fds to be an fps.

4.1. Linear fixed point systems

Finite dynamical systems over finite fields that are linear are very amenable to analysis and have been studied extensively in the literature (see [2, 9]).

In the multivariate case, a linear system over a finite field is represented by an fds (F_q^n, f) where f can be represented by an $n \times n$ matrix A over F_q. The fixed points of a multivariate fds (F_q^m, A) are simply the solutions to the homogeneous system of equations $(A - I)x = 0$.

In the finite field model for genetic networks, we assume that the number of states of each gene is a power of a prime. However, we will give a more general model that eliminates this assumption.

A *module* M over a ring R is said to be *finitely generated* if there exists a set of elements $\{s_1, s_2, \ldots, s_n\} \subset M$ such that $M = \{r_1 s_1 + r_2 s_2 + \cdots + r_n s_n \mid r_i \in R\}$. Finitely generated modules are generalized vector spaces. Examples are F_q^n and the set Z_m^n of n tuples over the ring of integers modulo an arbitrary integer m.

A *linear finite dynamical module system* (lfdms) consists of an ordered pair $(M(R), f)$ where $M(R)$ is a finitely generated module over a finite commutative ring R with unity and $f : M(R) \to M(R)$ is linear. Let $(M_1(R), f_1)$ and $(M_2(R), f_2)$ be lfdms. We define the *direct sum* of $(M_1(R), f_1)$ and $(M_2(R), f_2)$ to be the fds $(M_1 \oplus M_2, f_1 \oplus f_2)$ where $M_1 \oplus M_2$ is the direct sum of the modules $M_1(R)$ and $M_2(R)$ and $f_1 \oplus f_2 : M_1 \oplus M_2 \to M_1 \oplus M_2$ is defined by $(f_1 \oplus f_2)(u + v) = f_1(u) + f_2(v)$, for each $u \in M_1(R)$ and $v \in M_2(R)$. The state space of the direct sum is related to the component fds as follows.

Lemma 5. *Let G_1 be the state space of the lfdms $(M_1(R), f_1)$ and let G_2 be the state space of the lfdms $(M_2(R), f_2)$. Then the state space of the direct sum of $(M_1(R), f_1)$ and $(M_2(R), f_2)$ is $G_1 \times G_2$.*

This result has been noted in [9] for lfdms over fields.

We use the following well-known result (see, e.g., [19]) in order to establish necessary and sufficient conditions for an lfdms to be a fixed point system.

Lemma 6 (Fitting's lemma). *Let $(M(R), f)$ be an lfdms. Then there exist an integer $n > 0$ and submodules N and P satisfying*

(i) $N = f^n(M(R))$,
(ii) $P = f^{-n}(0)$,
(iii) $(M(R), f) = (N(R), f_1) \oplus (P(R), f_2)$, $f_1 = f|_N$ (the restriction of f to N) is invertible and $f_2 = f|_P$ is nilpotent.

Theorem 2. *Let $(M(R), f)$ be an lfdms and let N be defined as above. Then $(M(R), f)$ is a fixed point system if and only if either f is nilpotent or $f|_N$ is the identity map.*

Proof. By Fitting's lemma, we have $(M(R), f) = (N(R), f|_N) \oplus (P(R), f|_P)$ where $N = f^n(M(R))$ and $P = f^{-n}(0)$. Suppose that f is nilpotent. Then by Lemma 2, the state space of $(M(R), f)$ is a tree. Next suppose that $f|_N$ is the identity. Then by Lemma 3, the state space of $(M(N), f|_N)$ is a union of cycles each of length one and by Lemma 2, the state space of $(M(P), f|_P)$ is a tree. Hence by Lemma 4, the state space of $(M(R), f)$ is a union of trees and so $(M(R), f)$ is a fixed point system.

Conversely, suppose that $(M(R), f)$ is a fixed point system. Then the state space of $(M(R), f)$ is a union U of trees. If U consists of only one tree, then by Lemma 3, f is nilpotent. Now suppose that U is the union of at least two trees. Since f is invertible on N, it is also one-to-one on N. By Lemma 2, the state space of $(N(R), f|_N)$ is a union of cycles. Each of these cycles must be of length one. For if not, the state space of $(M(R), f)$ would contain at least one n-cycled tree where $n > 1$, contradicting that $(M(R), f)$ is a fixed point system. □

Theorem 2 can be used to prove the following result, which is suggested in [20, 21].

Corollary 1. *A linear finite dynamical system (F_q^n, f) over a field is a fixed point system if and only if the characteristic polynomial of f is of the form $x^{n_0}(x-1)^{n_1}$ and the minimal polynomial is of the form $x^{n_0'}(x-1)^{n_1'}$ where n_1' is either zero or one.*

Proof. Suppose (F_q^n, f) is an fps. Then either f is nilpotent or $f|_N$ is the identity. If f is nilpotent, then the characteristic polynomial of f is of the form x^{n_0} and the minimal polynomial of f is of the form $x^{n_0'}$. If $f|_N$ is the identity, then the characteristic polynomial of $f|_N$ is of the form $(x-1)^{n_1}$ and the minimal polynomial of $f|_N$ is of the form $(x-1)^{n_1'}$ where $0 \leq n_1' \leq n_1$. Furthermore, $n_1' \leq 1$ since otherwise (F_q^n, f) would not be an fps [2]. Therefore the characteristic and minimal polynomials of f are of the desired forms.

Conversely, suppose that the characteristic polynomial of f is of the form $x^{n_0}(x-1)^{n_1}$ and its minimal polynomial is of the form $x^{n_0'}(x-1)^{n_1'}$, where $n_0' \leq n_0$ and n_1' is either zero or one. If $n_0' = 0$, then the characteristic polynomial of f is $(x-1)^{n_1}$ and so the minimal polynomial of f is $x-1$, which implies that f is the identity and hence (F_q^n, f) is an fps. Next, suppose that $n_0' > 0$. Then either $n_1' = 0$ or $n_1' = 1$. If $n_1' = 0$, then the state space of (F_q^n, f) is a tree. If $n_1' = 1$, then the state space of (F_q^n, f) is the product of a tree and cycles of length one and, hence the union of trees. □

The corollary gives us a polynomial time algorithm to determine of a linear fds (F_q^n, f), where f is given by an $n \times n$ matrix, whether or not it is an fps. The characteristic polynomial of f can be determined in time $O(n^3)$ using the definition. The minimal polynomial of f can be determined in time $O(n^3)$ using an algorithm of Storjohann [22]. Both polynomials can be factored in subquadratic time using an algorithm of Kaltofen and Shoup [23].

4.2. Monomial systems

The simplest nonlinear multivariate fds (F_q^n, f) is one in which each component function f_i of f is a monomial, that is, a product of powers of the variables. In [24], Colón-Reyes et al. provide necessary and sufficient conditions that allow one to determine in polynomial time when an fds of the form (F_2^n, f), where f a monomial, is an fps. In [25], Colón-Reyes et al. give necessary and sufficient conditions for (F_q^n, f), where f is a monomial and q an arbitrary prime, to be an fps. However, one of these conditions is that a certain linear fds over a ring be an fps, but no criterion is given for such an fds to be an fps. Theorem 2 gives such a criterion. Let us describe the situation in more detail.

Definition 3. If $f = (f_1, f_2, \ldots, f_n)$ where each f_j is of the form $x_1^{\epsilon_{1j}} x_2^{\epsilon_{2j}} \cdots x_n^{\epsilon_{nj}}$, $j = 1, 2, \ldots, n$, where each ϵ_{ij} belongs to the ring Z_{q-1} of integers modulo $q-1$, then (F_q^n, f) is called a *monomial finite dynamical system*. The *log map* of (F_q^n, f) is defined by the $n \times n$ matrix $L_f = [\epsilon_{ij}]$, where $1 \leq i, j \leq n$. The *support map* is defined by $S_f = (h_1, h_2, \ldots, h_n)$ where each $h_i = x_1^{\delta_{i1}} x_2^{\delta_{i2}} \cdots x_n^{\delta_{in}}$ and where δ_{ij} is one if $\epsilon > 0$ and is zero otherwise.

The following theorem was published in [25].

Theorem 3 (Colón-Reyes, Jarrah, Laubenbacher, and Sturmfels). *A monomial fds (F_q^n, f) is an fps if and only if (Z_{q-1}^n, L_f) and (Z_2^n, S_f) are fixed point systems.*

Example 4. Consider the monomial fds (F_5^2, f) where $f = (xy, y) = (x^1y^1, x^0y^1)$. The matrix $L_f = \left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$ over Z_4 is nonsingular and hence not nilpotent. Furthermore, the n of Theorem 2 is $1, N = Z_4$, and L_f is not the identity. By Theorem 2, (Z_4, L_f) is not an fps and by the previous theorem, (F_5^2, f) is not an fps.

The problem of determining in polynomial time (in n) when an lfdms (R^n, f) is an fps, where R is a finite ring, is open.

4.3. A univariate approach

The fixed point problem is an important problem, suitable solutions for which have been obtained only in certain special cases. All of the work done so far has been done for multivariate fds. By considering the problem in the univariate domain, it is possible to gain some insight that is not evident in the multivariate domain. The results in the remainder of this section are examples of this.

Lemma 7. *(F_{q^n}, g) has fixed points if and only if $h(x) = \gcd(g(x) - x, x^{q^n} - x) \neq 1$ and in such a case, the fixed points are the zeros of $h(x)$.*

Proof. An element a of F_{q^n} is a fixed point of (F_{q^n}, g) if and only if a is a zero of $g(x) - x$. Since $x^{q^n} = x$ for all $x \in F_{q^n}$, $x^{q^n} - x$ contains all linear factors of the form $x - a$, $a \in F_{q^n}$ and so a is a zero of $g(x) - x$ if and only if $x - a$ is a factor of both $g(x) - x$ and $x^{q^n} - x$, that is, if and only if it is a factor of $h(x)$. □

Lemma 7 gives us algorithms for determining whether or not a given univariate fds has fixed points and if so, a method to find all such points. For the first part, we note that the greatest common divisor of two univariate polynomials of degree no more than d can be determined using no more than $(d \log^2 d)$ operations [26]. Since g has degree at most $q^n - 1$, this means that the complexity for calculating $h(x)$, that is, for determining whether or not a given univariate fds (F_{q^n}, g) is an fps $O(n^2 q^n)$.

When $h(x) \neq 1$, $h(x)$ can be factored in order to determine the set of all fixed points. At worst, using the algorithm in [23], the complexity of determining the factors of $h(x)$ is $O(d^{1.815} n)$, where d is the degree of $h(x)$. Clearly, d is less than or equal to the degree of $g(x)$, which in practice is determined by experimental data (e.g., from microarrays) and thus considerably less than the total number of possible points q^n. If we assume that the degree of $g(x)$ is not more than the square root of q^n, then $d^{1.815} n \leq n^2 q^n$ and the total complexity of the algorithm for determining all fixed points is thus $O(n^2 q^n)$.

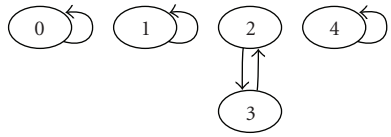

FIGURE 2: State space of (F_5, g), where $g(x) = x^3$ over F_5.

In contrast, the only known method for determining the fixed points of a multivariate fds (F_q^n, f) is the brute force method of enumerating all state transitions and for each value $f(a_1, a_2, \ldots, a_n)$ so generated, check to see if $f(a_1, a_2, \ldots, a_n) = (a_1, a_2, \ldots, a_n)$. The number of operations in this method is $O(q^{2n})$.

In many cases, the degree of $h(x)$ of Lemma 7 is small and its zeros can be found by inspection or by only several trials and errors. The lac operon example illustrates this.

Example 5. Let (F_{32}, g) be the fds describing the lac operon (Example 3). We have $h(x) = \gcd(g(x), x^{32} - x) = x^4 + \alpha^{26}x^3 + \alpha^{18}x^2 = x^2(x - \alpha^3)(x - \alpha^{15})$ and thus the fixed points are $x = 0$, $x = \alpha^3$, and $x = \alpha^{15}$.

Lemma 7 also gives a necessary condition for an fds to be an fps, which for emphasis we state as a theorem.

Theorem 4. *With the notation of Lemma 7, if (F_q, g) is an fps, then $h(x) \neq 1$.*

Proof. If $h(x) = 1$, then (F_{q^n}, g) has no fixed points and all cycles are nontrivial. Hence by Lemma 7, (F_{q^n}, g) is not an fps. □

The converse of Theorem 4 is not true.

Example 6. Consider the fds (F_5, g) where $g(x) = x^3$. Then $h(x) = \gcd(x^5 - x, x^3 - x) = x^3 - x \neq 1$, but (F_5, g) is not an fps. (see Figure 2).

5. IMPLEMENTATION ISSUES

One of the difficulties of implementing algorithms for the multivariate model is the choice of data structures, which can, in fact, affect complexity. For example, no algorithm is known for factoring multivariate polynomials that runs in time polynomial in the length of the "sparse" representation. However, such an algorithm exists for the "black box" representation (see, e.g., [27]).

On the other hand, data structures needed for algorithms for the univariate model are well known and simple to implement. In this case, one can also take advantage of well-known methods used in cryptography and coding theory. Table lookup methods for carrying out finite field arithmetic are an example. By using lookup tables we can make arithmetic operations at almost no cost. However, for very large fields, memory space becomes a limitation. Ferrer [28] has implemented table lookup arithmetic for fields of characteristic 2 on a Hewlett-Packard Itanium machine with two 900 MHz ia64 CPU modules and 4 GB of RAM. On this machine, we can create lookup tables of up to 2^{29} elements.

Multiplication is by far the most costly finite field operation and also the most often used, since other operations such as computing powers and computing inverses make use of multiplication. In other experiments on the Hewlett-Packard Itanium, Ferrer [28] takes advantage of machine hardware in order to implement a "direct" multiplication algorithm for F_{2^n} that runs in time linear in n for $n = 2$ up to $n = 63$ [28]. Here the field size is limited by the word-length of the computer architecture.

For larger fields, we can make use of "composite" fields (see, e.g., [29]), that is, fields F_n where n is composite, say $n = rs$. Making use of the isomorphism of $F_{2^{rs}}$ and $F_{(2^r)^s}$, we can use table lookup for a suitable "ground field" F_r and the direct method mentioned above for multiplication in the extension field $F_{(2^r)^s}$. Using the ground field F_{2^5} and selected values of s, Ferrer [28] obtains running time $O(s^2)$.

Still another approach to implement finite field arithmetic, that is, especially efficient for fields of characteristic 2, is the use of reconfigurable hardware or "field programmable gate arrays" (FPGAs). In [30], Ferrer, Moreno and the first author obtain a multiplication algorithm which outperforms all other known FPGA multiplication algorithms for fields of characteristic 2.

6. CONCLUSIONS

One piece of information that is of utmost interest when modeling biological events, in particular gene regulation networks, is when the dynamics reaches a steady state. If the modeling of such networks is done by discrete finite dynamical systems, such information is given by the fixed points of the underlying system. We have shown that we can choose between a multivariate and a univariate polynomial representation. Here we introduce a new tool, the discrete Fourier transform that helps us change from one representation to the other, without altering the dynamics of the system.

We provide a criterion to determine when a linear finite dynamical system over an arbitrary finitely generated module over a commutative ring with unity is a fixed point system. When a gene regulation network is modeled by a linear finite dynamical system we can then decide if such an event reaches a steady state using our results. When the finitely generated module is a finite field we can decide in polynomial time.

Gene regulation networks, as suggested in the literature, seem to obey very complex mechanisms whose rules appear to be of a nonlinear nature (see [31]). In this regard, we have made explicit some useful facts concerning fixed points and fixed point systems. We have given algorithms for determining when a univariate fds has at least one fixed point and how to find them. We have also given a necessary condition for a univariable fds to be a fixed point system. However, there are still much to be done and a number of open problems remain. In particular, what families of fds admit polynomial time algorithms for determining whether or not a given fds is an fps? This work is a first step towards the aim of designing theories and practical tools to tackle the general problem of fixed points in finite dynamical systems.

ACKNOWLEDGMENTS

This work was partially supported by Grant number S06-GM08102 NIH-MBRS (SCORE). The figures in this paper were created using the Discrete Visualizer of Dynamics software, from the Virginia Bioinformatics Institute (http://dvd.vbi.vt.edu/network_visualizer). The authors are grateful to Dr. Oscar Moreno for sharing his ideas on the univariate model and composite fields.

REFERENCES

[1] J. F. Lynch, "On the threshold of chaos in random Boolean cellular automata," *Random Structures & Algorithms*, vol. 6, no. 2-3, pp. 239–260, 1995.

[2] B. Elspas, "The theory of autonomous linear sequential networks," *IRE Transactions on Circuit Theory*, vol. 6, no. 1, pp. 45–60, 1959.

[3] J. Plantin, J. Gunnarsson, and R. Germundsson, "Symbolic algebraic discrete systems theory—applied to a fighter aircraft," in *Proceedings of the 34th IEEE Conference on Decision and Control*, vol. 2, pp. 1863–1864, New Orleans, La, USA, December 1995.

[4] D. Bollman, E. Orozco, and O. Moreno, "A parallel solution to reverse engineering genetic networks," in *Computational Science and Its Applications—ICCSA 2004—Part 3*, A. Laganà, M. L. Gavrilova, V. Kumar, et al., Eds., vol. 3045 of *Lecture Notes in Computer Science*, pp. 490–497, Springer, Berlin, Germany, 2004.

[5] A. S. Jarrah, H. Vastani, K. Duca, and R. Laubenbacher, "An optimal control problem for in vitro virus competition," in *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC')*, vol. 1, pp. 579–584, Nassau, Bahamas, December 2004.

[6] R. Laubenbacher and B. Stigler, "A computational algebra approach to the reverse engineering of gene regulatory networks," *Journal of Theoretical Biology*, vol. 229, no. 4, pp. 523–537, 2004.

[7] G. N. Fuller, C. H. Rhee, K. R. Hess, et al., "Reactivation of insulin-like growth factor binding protein 2 expression in glioblastoma multiforme," *Cancer Research*, vol. 59, no. 17, pp. 4228–4232, 1999.

[8] I. Shmulevich, E. R. Dougherty, and W. Zhang, "Gene perturbation and intervention in probabilistic Boolean networks," *Bioinformatics*, vol. 18, no. 10, pp. 1319–1331, 2002.

[9] R. A. Hernández Toledo, "Linear finite dynamical systems," *Communications in Algebra*, vol. 33, no. 9, pp. 2977–2989, 2005.

[10] J. Bähler and S. Svetina, "A logical circuit for the regulation of fission yeast growth modes," *Journal of Theoretical Biology*, vol. 237, no. 2, pp. 210–218, 2005.

[11] O. Moreno, D. Bollman, and M. Aviño, "Finite dynamical systems, linear automata, and finite fields," in *Proceedings of the WSEAS International Conference on System Science, Applied Mathematics and Computer Science, and Power Engineering Systems*, pp. 1481–1483, Copacabana, Rio de Janeiro, Brazil, October 2002.

[12] B. Sunar and D. Cyganski, "Comparison of bit and word level algorithms for evaluating unstructured functions over finite rings," in *Proceedings of the 7th International Workshop Cryptographic Hardware and Embedded Systems (CHES '05)*, J. R. Rao and B. Sunar, Eds., vol. 3659 of *Lecture Notes in Computer Science*, pp. 237–249, Edinburgh, UK, August-September 2005.

[13] M. Zivkovic, "A table of primitive binary polynomials," *Mathematics of Computation*, vol. 62, no. 205, pp. 385–386, 1994.

[14] R. E. Blahut, *Algebraic Methods for Signal Processing and Communications Coding*, Springer, New York, NY, USA, 1991.

[15] N. Yildirim and M. C. Mackey, "Feedback regulation in the lactose operon: a mathematical modeling study and comparison with experimental data," *Biophysical Journal*, vol. 84, no. 5, pp. 2841–2851, 2003.

[16] R. Laubenbacher, "Network Inference, with an application to yeast system biology," Presentation at the Center for Genomics Science, Cuernavaca, Mexico, September 2006, http://mitla.lcg.unam.mx/.

[17] R. Laubenbacher and B. Stigler, "Mathematical Tools for Systems Biology," http://people.mbi.ohio-state.edu/bstigler/sb-workshop.pdf.

[18] W. Just, "The steady state system problem is NP-hard even for monotone quadratic Boolean dynamical systems," submitted to *Annals of Combinatorics*.

[19] B. R. Macdonald, *Finite Rings with Identity*, Marcel Dekker, New York, NY, USA, 1974.

[20] O. Colón-Reyes, *Monomial dynamical systems*, Ph.D. thesis, Virginia Polytechnic Institute and State University, Blacksburg, Va, USA, 2005.

[21] O. Colón-Reyes, *Monomial Dynamical Systems over Finite Fields*, ProQuest, Ann Arbor, Mich, USA, 2005.

[22] A. Storjohann, "An $O(n^3)$ algorithm for the Frobenius normal form," in *Proceedings of the 23rd International Symposium on Symbolic and Algebraic Computation (ISSAC '98)*, pp. 101–104, Rostock, Germany, August 1998.

[23] E. Kaltofen and V. Shoup, "Subquadratic-time factoring of polynomials over finite fields," *Mathematics of Computation*, vol. 67, no. 223, pp. 1179–1197, 1998.

[24] O. Colón-Reyes, R. Laubenbacher, and B. Pareigis, "Boolean monomial dynamical systems," *Annals of Combinatorics*, vol. 8, no. 4, pp. 425–439, 2004.

[25] O. Colón-Reyes, A. S. Jarrah, R. Laubenbacher, and B. Sturmfels, "Monomial dynamical systems over finite fields," *Journal of Complex Systems*, vol. 16, no. 4, pp. 333–342, 2006.

[26] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison Wesley, Boston, Mass, USA, 1974.

[27] J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, Cambridge, UK, 2nd edition, 2003.

[28] E. Ferrer, "A co-design approach to the reverse engineering problem," CISE Ph.D. thesis proposal, University of Puerto Rico, Mayaguez, Puerto Rico, USA, 2006.

[29] E. Savas and C. K. Koc, "Efficient method for composite field arithmetic," Tech. Rep., Electrical and Computer Engineering, Oregon State University, Corvallis, Ore, USA, December 1999.

[30] E. Ferrer, D. Bollman, and O. Moreno, "Toward a solution of the reverse engineering problem usings FPGAs," in *Proceedings of the International Euro-Par Workshops*, Lehner, et al., Eds., vol. 4375 of *Lecture Notes in Computer Science*, pp. 301–309, Springer, Dresden, Germany, September 2006.

[31] R. Thomas, "Laws for the dynamics of regulatory networks," *International Journal of Developmental Biology*, vol. 42, no. 3, pp. 479–485, 1998.