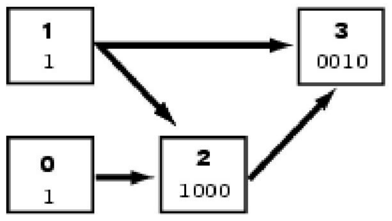
Error correction in genetic networks

Oscar Moreno, Carlos J. Corrada Bravo, Humberto Ortiz-Zuazaga, María Alicia Aviño-Diaz, and Dorothy Bollman.

Introduction: Dbnm and the reverse engineering problem.

In [3] Ideker, Thorsson and Karp describe a genetic network model based on Boolean networks, the deterministic Boolean network model or dBnm. In their model, gene levels or external biological stimuli can have two values (on or off). This is a greatly simplified biological model, which has the advantage of allowing simple computational analysis. Taking the model definition from this paper, we can describe a genetic network as:

1. A graph consisting of N numbered nodes and, $1 \leq n \leq N$.
2. A set of directed edges between nodes.
3. A Boolean function f_n for each node.
4. An edge from a node to another represents an influence of the first node on the expression of the second.



x_1	1	0	1	0
x_2	1	1	0	0
x_3	0	0	1	0

$$\begin{aligned} x_0 &:= 1 \\ x_1 &:= 1 \\ x_2 &:= x_0 \text{ and } x_1 \\ x_3 &:= x_1 \text{ and not } x_2 \end{aligned}$$

Figure 1: Example of the Boolean steady-state network model: (a) a directed graph structure with numbered nodes connected by edges, (b) the truth table (shown for node 3 only) and (c) the logic equations for each node.

Figure 1 reproduced from [3] represents a small example genetic network with 4 nodes and 4 edges. For a set of genes or stimuli and a set of perturbation experiments the authors construct an expression matrix as in Figure 2.

$$E = \begin{array}{|cccc|c} \multicolumn{1}{c}{x_0} & x_1 & x_2 & \multicolumn{1}{c}{x_3} & \\ 1 & 1 & 1 & 0 & p_0 \\ - & 1 & 0 & 1 & p_1 \\ 1 & - & 0 & 0 & p_2 \\ 1 & 1 & - & 1 & p_3 \\ 1 & 1 & 1 & + & p_4 \end{array}$$

Figure 2: Example expression matrix generated from the generic network in Figure 1.

The symbols + and- in the expression matrix represent genes or stimuli that are forced to 1 or 0, respectively. From this expression data, the challenge is to reconstruct or reverse engineer the genetic network. Ideker et al describe two inference procedures can reconstruct the genetic network interactively, suggesting new biological experiments to complement the existing data. From the expression matrix, the Predictor generates (possibly several) network hypothesis. The Chooser selects a new perturbation experiment, that would best discriminate between available hypotheses. Infering the network from the data in this manner means solving what is called the reverse engineering problem.

Our Definition of the dBnm.

We understand that the following is a formalization of the model presented in [3].

Definition 1. A Boolean variable assumes the values 0,1.

Definition 2. A Boolean function is a function involving Boolean variables and the operations and, or, not with the following definitions:

X	Y	X and Y	X or Y	Not X
1	1	1	1	0
1	0	0	1	0
0	1	0	1	1
0	0	0	0	1

Definition 3. A dBnm is a set of n Boolean variables $(x_1, \ldots, x_n)$, and a set of n Boolean functions $(f_1, \ldots, f_n)$. The Boolean variables represent genes or stimuli, and the Boolean function f_i represents how the gene i is determined by all the other genes.

Lemma 1 *Given n Boolean variables $(x_1, \ldots, x_n)$ and define $f(x_1, \ldots, x_n)$ for all possible values. Then there is a Boolean function that coincides with f as defined.*

Proof: See any book on computer architecture ([9]) for realizing a Boolean function as sums of products and products of sums.

Deterministic Finite Field Network Models with Stimuli

In [8] a finite state system is defined to study the behavior of various types of systems as an idealized model for a large number of phenomena.

Definition 4. A (complete, deterministic) finite state system[1] M is defined as follows:

1. A finite, nonempty set $U = \{\alpha_1, \alpha_2, \ldots, \alpha_h\}$, called the input alphabet of M. An element of U is called an input symbol.
2. A finite, nonempty set $Y = \{\beta_1, \beta_2, \ldots, \beta_s\}$, called the output alphabet of M. An element of Y is called an output symbol.
3. A finite, nonempty set $S = \{\sigma_1, \sigma_2, \ldots, \sigma_r\}$, called the state set of M. An element of S is called a state.
4. A next-state function f that maps the set of all ordered pairs (σ_i, α_j) into S.
5. An output function g that maps the set of all ordered pairs (σ_i, α_j) into Y.

A generalization of a FSS where the input and output alphabets carry the structure of a vector space over a finite field.

Definition 5. A modular system (MS) M of order n over GF(q) is defined by the following:

1. a $k-$dimensional vector space U over GF(q), called input space of M, the elements of which are called inputs and are written as column vectors.
2. An $m-$dimensional vector space Y over GF(q), called output space of M, the elements of which are called outputs and are written as column vectors.
3. An $n-$dimensional vector space S over GF(q), called state space of M, the elements of which are called states and are written as column vectors.

[1]In the computer science literature a finite state system is called a Mealy machine and finite state system with $Y = \phi$ is called a finite automata.

If the next-state and output functions are linear it is called a linear MS (LMS) and the following must hold:

1. Four characterizing matrices over GF(q):

$$A = (a_{ij})_{n\times m}, B = (b_{ij})_{n\times k}, C = (c_{ij})_{m\times n}, D = (d_{ij})_{m\times k}.$$

   The matrix A is called the characteristic matrix of M.

2. A rule relating the state at time $t + 1$ and output at time t to the state and input at time t:

$$\mathbf{s}(t+1) = A\mathbf{s}(t) + B\mathbf{u}(t), \mathbf{y}(t) = C\mathbf{s}(t) + D\mathbf{u}(t).$$

Using this definition and the definition of dFGnm in [20] we can defined a deterministic finite field genetic network model with stimuli as follows:

Definition 6. A Deterministic Finite Field Network Models with Stimuli (dFGnms) is a MS of order n over GF(q) where the stimuli determines the input space U and $S = \{(\gamma_1, \gamma_2, \ldots, \gamma_n)^T \mid \gamma_i \text{ are the values of the gene } i\}$. The rest follows the definition of dFGnm in [20].

Note that a deterministic boolean network model with stimuli (dBnms) is a specific case of a dFGnms where $q = 2$.

An example of a dBnms can be constructed from the example shown in Figure 1, with the set of stimuli define as $\{p_0, p_1, p_2, p_3, p_4\}$ where p_1 changes x_0, p_2 changes x_1, p_3 changes x_2 and p_4 changes x_3 as in the table of Figure 2. This table shows an example where the first state is $(1, 1, 1, 0)$ but the same method can be follow in general.

Error Correction in Genetic Networks.

Our finite field deterministic genetic network model from previous sections can be endowed with error correcting capabilities in the following manner. First, from the biological data we inferred the model that fits this data using the method described in previous sections. It was also proven that this model is a finite dynamical system. Once the finite dynamical system (also called finite state machine or linear automata) has been inferred, we can provide the error correction by constructing a trellis diagram (see Example 1).

We will use the Viterbi algorithm to find the path that the process took with maximum likelihood. This algorithm, introduced in [5] for the decoding of convolutional codes, was proven ([6, 7]) to select the path that gives the largest value of the likelihood function. In general it can be used in any system where the next state depends on past states, like a hidden Markov chain. A version of this algorithm that outputs the probabilities of the paths instead of only the best path (called soft-Viterbi) will also be useful for future research of our continuous model described in a previous section.

Example 1. In the following figures we can generate a machine and its trellis from an MS. This is done by expanding the state diagram in time, representing each time unit with a separate state diagram.

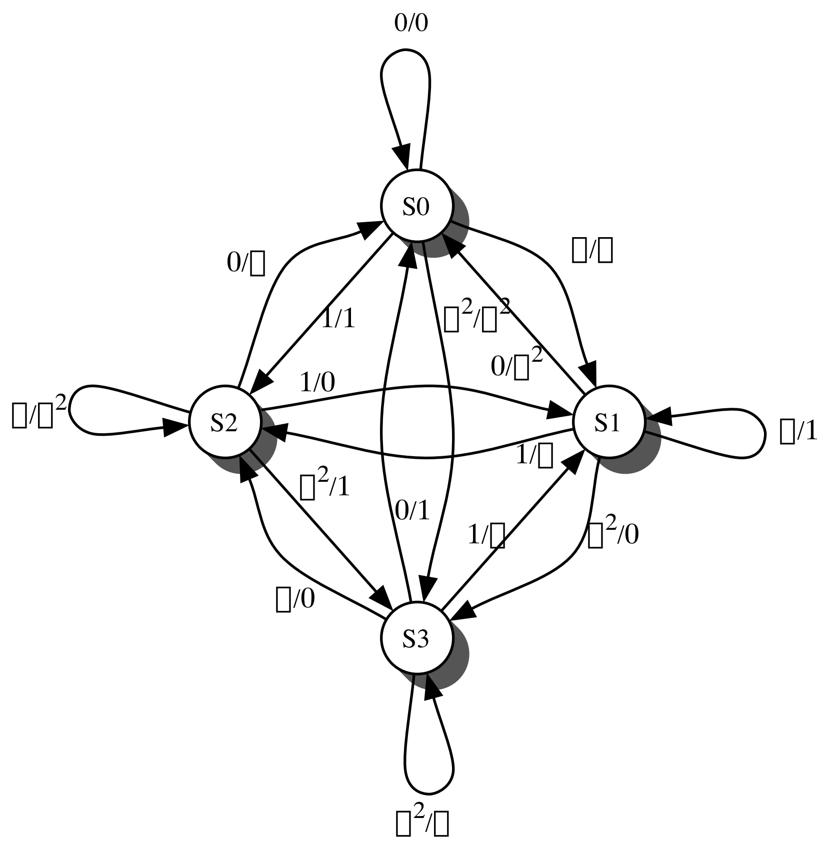


Figure 3: Finite State Machine or FDS from an MS

Acknowledgment

We want to acknowledge the inspiration of our error correcting scheme comes from the work of Prof. Solomon Golomb in [11].

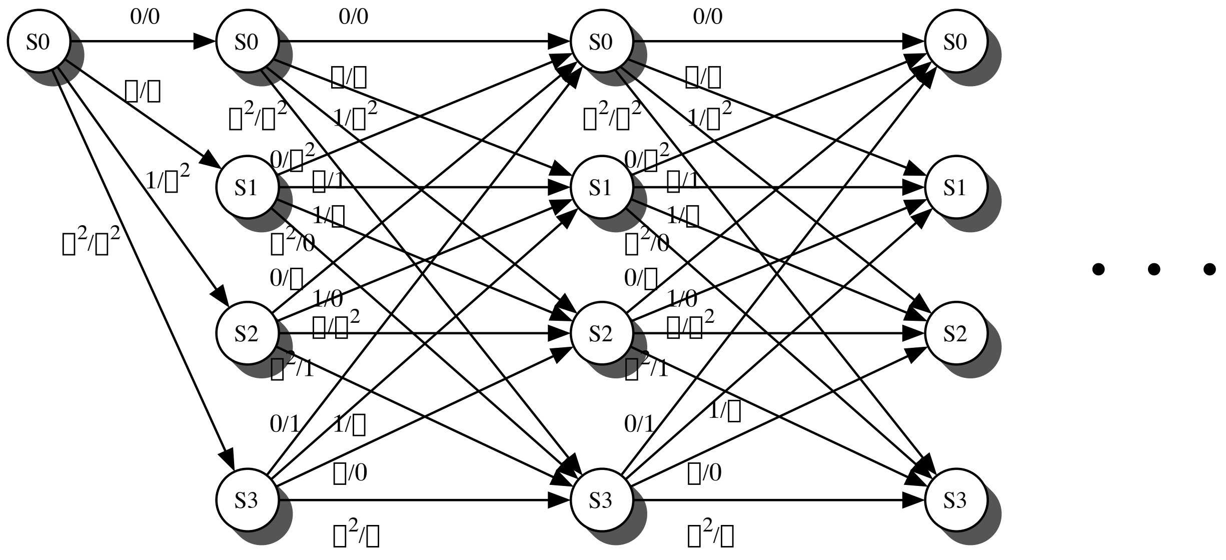


Figure 4: Trellis for error correction in the machine of Figure 3.

References

[1] T. Akutsu, S. Kuahara, O. Maruyama, S. Miyano, Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions. Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms, H. Karloff, ed. (ACM Press, 1998).

[2] R. Somogyi and C. Sniegoski, Modeling the complexity of genetic networks: Understanding multigenic and pleiotropic regulation. Complexity 1, 45-63 (1996).

[3] T.E. Ideker, V. Thorsson, and R.M. Karp. Discovery of Regulatory Interactions Through Perturbation: Inference and Experimental Design, Pacific Symposium on Biocomputing 5:302-313 (2000).

[4] O. Moreno, D. Bollman, M. Aviño-Diaz, Finite Dynamical Systems, Linear Automata, and Finite Fields, Conference Proceedings, WSEAS Transactions, (2002).

[5] A. J. Viterbi, Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm, IEEE Trans. Inf. Theory, 260-269, April 1967.

[6] G. D. Forney, Jr., The Viterbi Algorithm, Proc. IEEE, 268-278, March 1973.

[7] G. D. Forney, Jr., Convolutional Codes II: Maximum Likelihood Decoding,Inf. Control, 222-266, July 1974.

[8] R. Lidl and H. Niederreiter, Finite Fields, Cambridge University Press, 1997.

[9] D. A. Patterson and J.L. Hennessy, Computer organization and design, Morgan Kaufmann Publishers, San Francisco 1997

[10] S. Golomb and E.C. Posner. Rook Domains, Latin Squares, Affine Planes, and Error-Distributing Codes. IEEE Transaction and Information Theory, July 1964, IT-10, 116-118.

[11] S. Golomb. Error-correcting codes and the genome project. Computers Chem. Vol 16, No 2 pp 183-186, 1992.

[12] S. Golomb, Genetic coding. Eng. And Sci., pp 9-14. April 1962.

[13] C. Barret, and C. Reidys, Elements of a theory of simulation I: Sequential CA Over Random Graphs, Appl. Math. Comput. 98pp 241-259, (1999).

[14] C. Barret, H. Mortveit, and C. Reidys, Elements of a theory of simulation II: sequential dynamical systems, Appl. Math. Comput. 107 (2000).

[15] C. Barret, H. Mortveit, and C. Reidys, Elements of a theory of simulation III: Equivalence of SDS, Appl. Math. Comput. 122 pp 325-340,(2001).

[16] D. Bollman, Some periodicity properties of transformations on vector spaces over residue class rings, J. Soc. Indust. Appl. Math. Vol 13, No3, September, 1965, pp 902-912.

[17] R. Laubenbacher and B. Pareigis, Equivalence relations on finite dynamical systems. Adv, in Appl. Math 26 (2001), 237-251.

[18] R. Laubenbacher and B. Pareigis, Decomposition and simulation of sequential dynamical systems, Preprint, (2002).

[19] R. Laubenbacher and B. Pareigis, Update schedules of Sequential Dynamical Systems preprint, (2002).

[20] A finite field deterministic genetic network model. submitted to the AAECC 2003.