Pacific Symposium on Biocomputing 3:18-29 (1998)

REVEAL, A GENERAL REVERSE ENGINEERING ALGORITHM FOR INFERENCE OF GENETIC NETWORK ARCHITECTURES

SHOUDAN LIANG

SETI Institute, NASA Ames Research Center,
Moffett Field, CA 94035 (sliang@mail.arc.nasa.gov)

STEFANIE FUHRMAN, ROLAND SOMOGYI

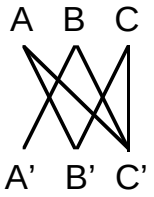
Molecular Physiology of CNS Development, LNP/NINDS/NIH, 36/2C02, Bethesda, MD 20892 (http://rsb.info.nih.gov/mol-physiol/homepage.html; sfuhrman@codon.nih.gov; rolands@helix.nih.gov)

Given the immanent gene expression mapping covering whole genomes during development, health and disease, we seek computational methods to maximize functional inference from such large data sets. Is it possible, in principle, to completely infer a complex regulatory network architecture from input/output patterns of its variables? We investigated this possibility using binary models of genetic networks. Trajectories, or state transition tables of Boolean nets, resemble time series of gene expression. By systematically analyzing the mutual information between input states and output states, one is able to infer the sets of input elements controlling each element or gene in the network. This process is unequivocal and exact for complete state transition tables. We implemented this REVerse Engineering ALgorithm (REVEAL) in a C program, and found the problem to be tractable within the conditions tested so far. For n=50 (elements) and k=3 (inputs per element), the analysis of incomplete state transition tables (100 state transition pairs out of a possible 10^{15}) reliably produced the original rule and wiring sets. While this study is limited to synchronous Boolean networks, the algorithm is generalizable to include multi-state models, essentially allowing direct application to realistic biological data sets. The ability to adequately solve the inverse problem may enable in-depth analysis of complex dynamic systems in biology and other fields.

Binary models of genetic networks

Virtually all molecular and cellular signaling processes involve several inputs and outputs, forming a complex feedback network. The information for the construction and maintenance of this signaling system is stored in the genome. The DNA

a



b

A' = B
B' = A or C
C' = (A and B) or (B and C) or (A and C)

c

input			output		
A	**B**	**C**	**A'**	**B'**	**C'**
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	1
1	0	0	0	1	0
1	0	1	0	1	1
1	1	0	1	1	1
1	1	1	1	1	1

Fig. 1 A simple Boolean network. a) Wiring diagram. b) Logical (Boolean) rules. c) Complete state transition table defining network. The input column corresponds to the state at time=t, the output column (elements marked by prime) corresponds to the state at time=t+1.

sequence codes for the structure and molecular dynamics of RNA and proteins, in turn determining biochemical recognition or signaling processes. The regulatory molecules that control the expression of genes are themselves the products of other genes. Effectively, genes turn each other *on* and *off* within a *proximal genetic network* of transcriptional regulators (Somogyi and Sniegoski, 1996). Furthermore, complex webs involving various intra- and extracellular signaling systems on the one hand depend on the expression of the genes that encode them, and on the other hand control the expression of genes as the signals terminate at transcriptional regulation. All in all, the information stored in the DNA determines the dynamics of the *extended genetic network*, the state of which at a particular time point should be reflected in gene expression patterns (Somogyi and Sniegoski, 1996). We have developed the basic tools to measure these gene expression patterns, and are now concerned with inferring the functional network architectures from time series or state transition sets (Somogyi et al., 1996).

A rational approach to designing genetic network analysis tools is based on generating model systems on which the performance of the tools can be tested. The simplest such model system is the Boolean network. Genes correspond to elements in a Boolean net, the wiring of the elements to one another correspond to functional links between genes, and the rules determine the result of a signaling interaction given a set of input values. Genes are idealized as being either *on* or *off*, resulting in binary elements interacting according to Boolean rules. Given a particular set of elements, wiring, rules, a particular trajectory or the state transition table covering all trajectories of a network can be calculated (Fig. 1). Such a trajectory must reach a final repeating state cycle, for the simple reason that the network only has 2^n states, and each state transition is unequivocally determined (after maximally 2^n iterations, a repeating state must be found). An attractor may be a single state (*point attractor*, corresponding to a “steady state”) or may comprise several states (*dynamic attractor*,

corresponding to a "limit cycle"). Attractors may be envisioned as the "target area" of an organism, e.g. cell types at the end of development, repaired tissue following a response to injury, or even adaptation of metabolic gene expression following a change in nutrient environment in bacteria (see Kauffman, 1993; Somogyi and Sniegoski, 1996; Wuensche, 1992).

Testing of algorithms for extracting network architectures from state transition measurements will require knowledge of the original network that was to be inferred. This is not yet possible with living systems. Using Boolean networks, we can facilely generate model state transition tables. Depending on the assumptions we make about living genetic networks regarding size, connectivity, redundancy and complexity, we can simulate these conditions in Boolean nets, and test how well our reverse engineering procedure works against these different backdrops. Given good results, our confidence in this approach should be warranted. Below we will use systematic mutual information analysis of Boolean network state transition tables to extract minimal network architectures.

Information theoretic principles of mutual information (M) analysis

Information theory provides us with a quantitative information measure, the Shannon entropy, H. The Shannon entropy is defined in terms of the probability of observing a particular symbol or event, p_i, within a given sequence (Shannon & Weaver, 1963),

$$H= - \Sigma\ p_i \log p_i.$$

A few illustrations (Figs. 2 & 3) of a binary system shall help explain the behavior of H. In a binary system, an element, X, may be in either of s=2 states, say *on* or *off*. Over a particular sequence of events (Fig. 2a), the sum of the probabilities of X being *on*, p(1) or *off*, p(0) must be equal to unity, therefore p(1)=1-p(0), and H(X)=-p(0)*log[p(0)]-[1-p(0)] *log[1-p(0)]. H reaches its maximum when the *on* and *off* states are equiprobable (Fig. 3a), i.e. the system is using each information carrying state to its fullest possible extent. As one state becomes more probable than the other, H decreases - the system is becoming biased. In the limiting case, where one probability is unity (certainty) and the other(s) zero (impossibility), H is zero (no uncertainty - no freedom of choice - no information).

The maximum entropy, H_{max}, occurs when all states are equiprobable, i.e. p(0)=p(1) =1/2. Accordingly,

$$H_{max}=\log(2).$$

Entropies are commonly measured in "bits" (binary digits), when using the logarithm on base 2; e.g. $H_{max}=1$ for a 2 state system.

Our aim is to compare different sequences using information measures to establish functional relationships between elements of a network. In a system of 2 binary elements, X (index i) and Y (index j), the individual and combined Shannon entropies are defined essentially as above (Fig. 2b):

$H(X) = -\Sigma\, p_i \log p_i$,
$H(Y) = -\Sigma\, p_j \log p_j$, and
$H(X, Y) = -\Sigma\, p_{i,j} \log p_{i,j}$

There are 2 conditional entropies which capture the relationship between the sequences of X and Y, H(X|Y) and H(Y|X). These are related as follows (Shannon & Weaver, 1963):

$$H(X,Y) = H(Y|X) + H(X) = H(X|Y) + H(Y) .$$

In words, the uncertainty of X and the remaining uncertainty of Y given knowledge of X, H(Y|X), i.e. the information contained in Y that is not shared with X, sum to the entropy of the combination of X and Y.

We can now find an expression for the shared or "mutual information", M(X,Y), also referred to as "rate of transmission" between an input/output channel pair (Shannon & Weaver, 1963):

$$M(X,Y) = H(Y) - H(Y|X) = H(X) - H(X|Y).$$

The shared information between X and Y corresponds to the remaining information of X

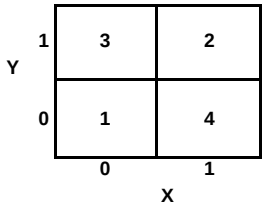
a

X	0	1	1	1	1	1	1	0	0	0
Y	0	0	0	1	1	0	0	1	1	1

H(X) = -0.4log(0.4)-0.6log(0.6) = 0.97 *(40% 0s and 60% 1s)*

H(Y) = -0.5log(0.5)-0.5log(0.5) = 1.00 *(50% 0s and 50% 1s)*

b



H(X,Y) = -0.1log(0.1)-0.4log(0.4)-0.3log(0.3)-0.2log(0.2) = 1.85

Fig. 2 Determination of H. a) Single element. Probabilities are calculated from frequency of *on/off* values of X and Y. b) Distribution of value pairs. H is calculated from the probabilities of co-occurrence.

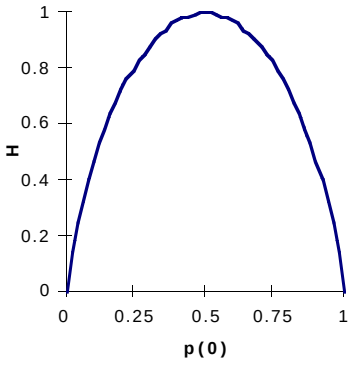


Fig. 3 Shannon entropies for a 2-state information source. Since the sum of the state probabilities must be unity, p(1)=1-p(0) for 2 states.

if we remove the information of X that is not shared with Y. Using the above equations, mutual information can be defined directly in terms of the original entropies; this formulation will be important for the considerations below:

$$M(X,Y) = H(X) + H(Y) - H(X,Y).$$

The Venn diagrams of Fig. 4 illustrate the relationships between these measures. We will use these information principles to extract the critical connections between network elements from binary network state transition data. Similar analyses have been previously explored in the classification of Boolean rules (Somogyi & Fuhrman, 1997) and genotype-phenotype mappings (Sawhill, 1995).

The core of REVEAL: systematic M-analysis of state transition tables

The general strategy of our algorithm is to use mutual information measures to extract the wiring relationships from state transition tables. These directly lead to the look-up tables of the rules. We shall explain the step-by-step workings of the algorithm (Fig. 5) in the analysis of the network example of Fig. 1.

Step 1: Identification of k=1 links

We begin by determining the pair-wise mutual information matrix (Fig. 5) of all single input-output pairs (Fig. 1). A "prime" denotes the output state of an element, e.g. A'. If M(A',X)=H(A'), i.e. M(A',X)/H(A')=1, then X exactly determines A'. This is the case for B and A' in Fig. 5.

Note: Since H(A')=M(A',X), then

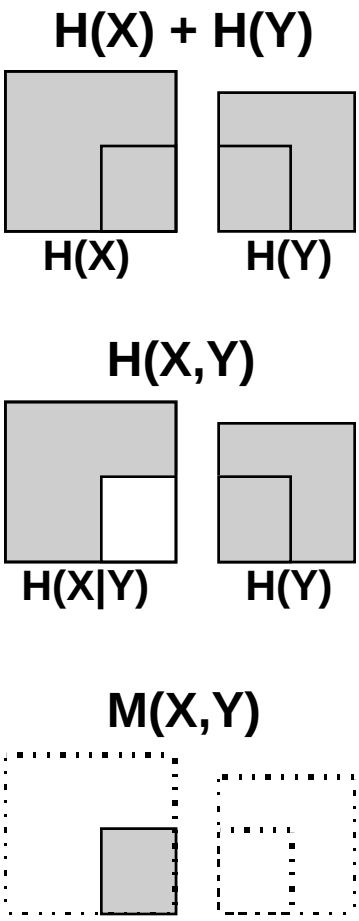


Fig. 4 Venn diagrams of information relationships. In each case, add the shaded portions of both squares to determine one of the following: [H(X)+H(Y)], H(X,Y), and M(X,Y). The small corner rectangles represent information that X and Y have in common. H(Y) is shown smaller than H(X) and with the corner rectangle on the left instead of the right to indicate that X and Y are different, although they have some mutual information.

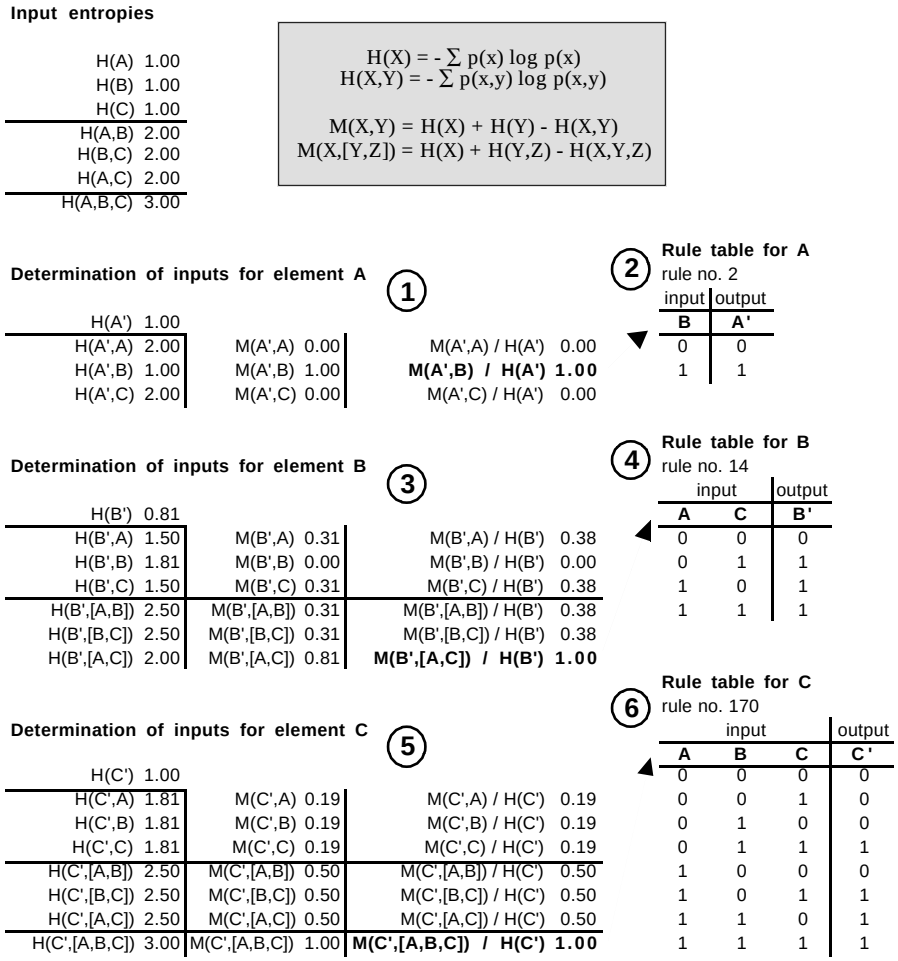


Fig. 5 Outline of progressive M-analysis underlying REVEAL for network example shown in Fig. 1. Hs and Ms are calculated from the look-up tables according to the definitions (shaded). The network wiring is extracted by M-analysis (left, odd steps). Rule tables are then determined directly from the trajectory (right, even steps).

H(X)=H(A',X), i.e. it is not even necessary to calculate M(A',X) explicitly, making the computation marginally faster. The measurement of M must be sufficiently precise in the analysis of many state transition pairs, i.e. the determination of p (the probability) must be able to distinguish a change of 1/T (T=number of state transition pairs).

Step 2: Determination of k=1 rule

The look-up table for the input/output pairs (Fig. 1) constitutes the rule. Redundant input/output pair listings are eliminated in the proper rule table format (Fig. 5).

Step 3: Identification of k=2 links

If not all state transitions can be explained in terms of k=1, we will determine entropies of input pair combinations with the remaining unsolved output elements. If M(B',[X,Y])=H(B'), or, more concisely, H(B',X,Y)=H(X,Y), then the pair [X,Y] completely determines B'. In Fig. 5, no single input can predict B', but the pair [A, C] accounts for all of the entropy of B'.

Step 4: Determination of k=2 rule (as above for k=1)

Step 5: Identification of k=3 links

If not all elements can be resolved in terms of k=1 and k=2, the next step is to determine the entropies of input triplet combinations with the remaining unsolved output elements. In our example (Fig. 5), since M(C',[A,B,C])=H(C'), or, more concisely, H(C',A,B,C)=H(A,B,C), then [A,B,C] completely determines C'.

Step 4: Determination of k=3 rule (as above for k=1)

Step i: Identification of k=i links (i<=n, number of network elements).

This applies to networks of any size. If not all trajectories can be explained in terms of k=i-1, i-2 . . . 1 inputs, the search pursues the entropies of i-let input combinations with the remaining unsolved output elements. If M(Y, [X_1, X_2, X_3,. . . X_i])=H(Y), or H(Y,X_1, X_2, X_3,. . . X_i)= H(X_1, X_2, X_3,. . . X_i) then the i-let, [X_1,X_2,X_3,. . .X_i] completely determines Y (Y=output element). Naturally, the input value combinations of the combined state transition tables covering Y and [X_1,X_2,X_3,. . .X_i] define the look-up table of the rule.

The advantage of this algorithm is that simple networks can be calculated very quickly just by comparing Hs of state transition pairs. The algorithm will calculate the Hs for higher k only as required. Of course, as k increases, the calculations of the Hs will require progressively more time (see below). The goal is obviously to minimize the number of computationally intensive operations. We are currently exploring rational search optimization procedures (e.g. minimization of k combination testing) based on probable rule restrictions. Moreover, REVEAL is amenable to parallel computing, which we are planning to pursue in the future.

Implementation of the algorithm

A practical implementation of REVEAL will require careful considerations regarding non-redundant and biologically feasible rule inference. Amongst $2^{(2^k)}$ input rules, many do not depend on one or more of their inputs. These rules are equivalent to a rule with a smaller k_{eff} (*effective* k; Somogyi & Fuhrman, 1997). Since we can only detect rules that truly depend on all of their inputs, the best we can do is to infer the

equivalent rule with minimum k_{eff}. For this reason, the k-input rules we used in constructing test networks are effective k-input rules, i.e. that they cannot be reduced to a rule with smaller number of inputs.

There are two one-input (k=1) and ten two-input (k=2) rules that truly depend on all their inputs. Two of the ten two-input rules, *exclusive or* and *equivalent*, may be unlikely to occur biologically. They produce minimally correlated behavior in networks (atypical for biological networks), and would be difficult to encode in biomolecular interactions. One may consider eliminating such rules from biologically feasible test networks.

For k=3 rules, there are 218 rules of k_{eff}=3, 30 of k_{eff}=2, 6 of k_{eff}=1 and 2 of k_{eff}=0. Of course, we limit the construction of model networks to rules of an effective k. Moreover, k=3 rules may be further restricted according to biological plausibility (as above for k=2).

In order to infer the rule for a particular gene, our strategy is to first test if it is an effective one-input rule. Since the input for the gene could be anywhere in the network, there are *N* possible inputs for each gene. Each one is tested in turn using the mutual information analysis discussed earlier. For genes whose output is not determined solely by any one input, the effective k for the rule of that gene is larger than one. We next determine whether the gene is determined by a rule with two effective inputs. There are $\frac{N(N-1)}{2}$ pairs of possible inputs for a two-input rule. For each of the these input pairs, we use the M-analysis to determine whether the input pair specifies the output value for the gene. In general we have $\binom{N}{k} = \frac{N!}{k!(N-k)}$ possible inputs for a k-input rule. All of the $\binom{N}{k}$ input combinations are examined to find the correct input set.

Performance of Algorithm

In principle, the information theoretic approach requires computation of the probability over all 2^N state transition pairs of the network for each of the $\binom{N}{k}$ possible wirings of the k-input rule. For a network of moderate size (*N*=50), the number of configurations becomes too large to compute. Fortunately, the criterion for determining the causality relationship in the M-analysis is satisfied using any finite input set provided that the same set is used in computing all quantities involved. For example, the criterion (see above) for determining that C' is the output of A and B is H([A,B],C')=H(A,B). A finite set of randomly selected input patterns [A,B] can be used to construct a 4-bit histogram. From the histogram, we

obtain the probabilities needed to compute H(A,B). H([A,B],C') is computed from an 8-bit histogram using the same set of input patterns combined with the output C'. If [A,B] is the correct input for C', then H([A,B],C')=H(A,B) will be satisfied for any input set.

However, if the number of input-output pairs is small, there will be a large number of mis-identifications because of incidental degeneracy. In order to estimate the size of the sample set needed to uniquely identify the right wiring for a gene, we compute the probability of mis-identification as a function of increasing the sample size, *S*, which is defined as the number of input-output pairs used in computing the histogram. The probability is computed by counting the number of input wirings that satisfy the M-analysis criterion normalized to all possible $\binom{N}{k}$ wirings for a k-input rule. Fig. 6 shows that the probability declines exponentially with increasing *S*. With a very small *S*, much smaller than the total number of all possible state transition patterns 2^N, we can already identify the correct wiring.

The networks used in testing REVEAL are constructed using a mixture of

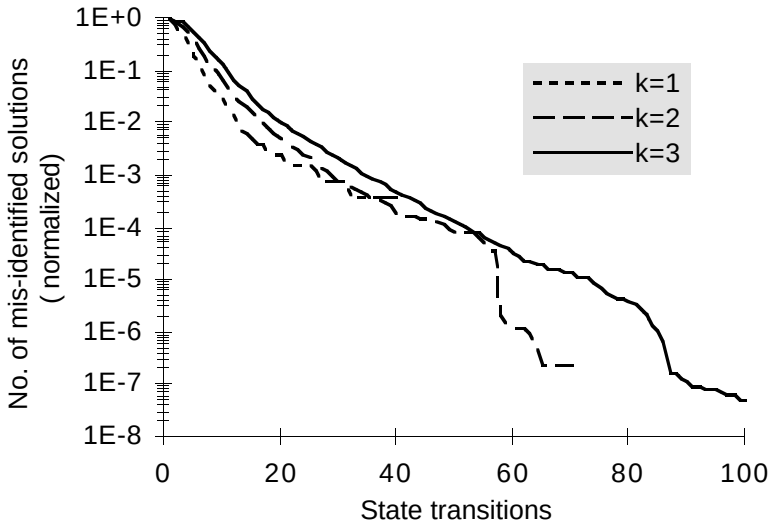


Fig. 6 Reduction of mis-identified network wiring solutions. The number of erroneous wirings identified by the M-analysis (normalized) versus the number of state transition pairs used for *effective* k value k=1,2,3. The data was obtained by averaging over 50 random wirings for a network with 50 elements. Note that a correct solution is always found; this is subtracted from the plotted number of solutions.

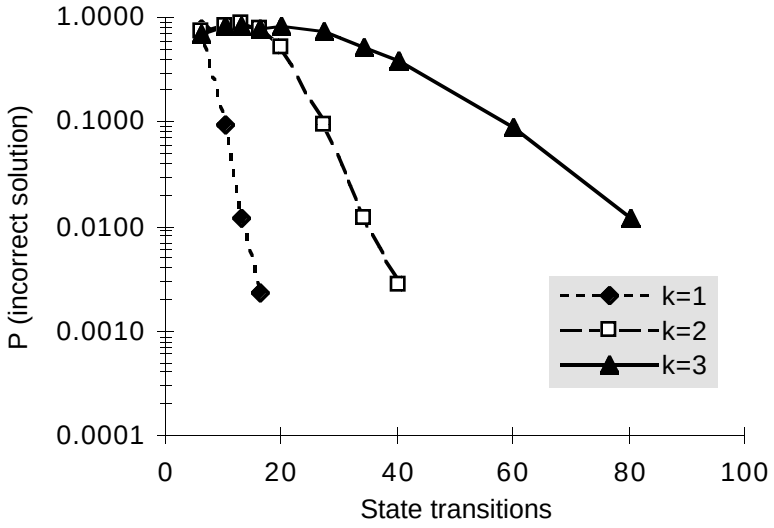


Fig. 7 Convergence of solution in random network. The probability of not finding the perfect solution, *P*, versus the number of state transition pairs used, *S*, for effective k value k=1,2,3. Each data point is computed by averaging over 150 random wirings for a network with 50 elements. The network is constructed with one third each of one-input, two-input and three-input rules. For each of the three cases, the rule selection is made at random amongst all the effective k-input rules. As more transition pairs are used, the probabilities decay exponentially at large S after a relatively flat plateau. Our data also indicate that P becomes zero at *S*=100 for k=3; at *S*=60 for k=2; and at *S*=20 for k=1 (not graphically depictable on log scale).

one-input, two-input, and three-input rules with equal probability. When a gene is assigned a k-input rule, one of k-input rules is selected for the gene at random from all eligible rules. In the case of Fig. 7, all the rules that truly depend on all their inputs are eligible. There are 2 such one-input rules, 10 such two-input rules and 218 such three-input rules.

Every rule assigned to genes has been correctly identified for all 150 networks used for Fig. 7. In the most difficult case of k_{eff}=3, all the rules have been uniquely identified (perfect solution) when the number of state transition pairs reaches 100. For one-input and two-input rules, the perfect solution is reached when *S*=20 and *S*=60 respectively for all the genes in 150 networks. When S is smaller than these limit values, some genes are allocated more than one set of inputs by the M-analysis, i.e. there is more than one solution. The number of degenerate solutions as a function of the number of state transition pairs was discussed in Fig. 6.

Outlook

We have shown that REVEAL performs well for networks of low k (number of inputs per gene). For higher k, the algorithm should be accelerated through a) parallelization, and b) increasing the search efficiency of solution space, e.g. by taking maximal advantage of wiring and rule constraints. We are currently pursuing these strategies.

Boolean networks are based on the notion that biological networks can be represented by binary, synchronously updating switching networks. In real biological systems, however variables change continuously in time. This behavior can be approximated by asynchronous Boolean networks (reviewed in Thieffry & Thomas, 1998), or continuous differential equations that capture the structure of logical switching networks (Glass, 1975). The issue of determining the logical structure of a continuous network based on knowledge of the transitions was explicitly addressed in a previous work on oscillating neural networks (Glass and Young, 1979). The point of REVEAL is to base causal inference on the most fundamental and general correlation measure available, mutual information. While we concentrated on idealized Boolean networks, mutual information measures can be applied to multivalued discrete and also continuous data sets. Of course, once multiple states are introduced, corresponding flexibility will also be found in the timing. Since continuous behavior can be approximated by discrete systems given sufficient resolution, REVEAL could be applied to appropriately discretized continuous data sets. However, the introduction of multiple states will greatly increase the number of theoretically possible state transitions; network and wiring constraints must therefore be carefully considered when generalizing REVEAL to multivalued networks. For example, integration of cluster analysis for the inference of shared inputs (currently applied to continuous, large scale gene expression data sets; see Michaels et al., 1998) could quickly identify wiring constraints and simplify the overall inference process.

Finally, as REVEAL or potential successors become more refined, we need to consider the data sets that must be generated to allow maximal depth of inference. The algorithm relies on the analysis of state transitions or temporal responses of gene expression patterns (or other relevant biological parameters!) to perturbations or internal changes (e.g. development). What will be the proper time step across which measurements need to be acquired and interpreted? How many perturbations will be necessary to capture sufficient diversity? How many states (if more than binary) need to be attributed to each biological parameter? The potential rewards of fundamental insights into genetic and biological signaling networks should encourage us to pursue these questions.

Acknowledgments

S.L. acknowledges the support of NASA Cooperative Agreement NCC2-974. We thank Ian McDonald, Leon Glass and Patrick D'haeseleer for helpful comments.

References

Glass, L. (1975). Classification of Biological Networks by Their Qualitative Dynamics. *J. Theor. Biol.* 54:85-107.

Glass, L. and Young, R. (1979) Structure and Dynamics of Neural Network Oscillators. *Brain Res.* 179:207-218.

Kauffman, S.A. (1993) The Origins of Order, Self-Organization and Selection in Evolution. Oxford University Press.

Michaels G., Carr D.B., Wen X., Fuhrman S., Askenazi M. and Somogyi R. (1998) Cluster Analysis and Data Visualization of Large-Scale Gene Expression Data. *Proc. Pacific Symposium on Biocomputing 1998*. In press.

Sawhill, B.K. (1995) Genetic Function Analysis. *Santa Fe Institute Working Paper* 95-10-085.

Shannon, C.E. and Weaver, W. (1963) The Mathematical Theory of Communication. University of Illinois Press.

Somogyi, R. and Fuhrman, S. (1997) Distributivity, a general information theoretic network measure, or why the whole is more than the sum of its parts. *Proc. International Workshop on Information Processing in Cells and Tissues (IPCAT) 1997*. In press.

Somogyi R. and Sniegoski C.A. (1996) Modeling the Complexity of Genetic Networks: Understanding Multigenic and Pleiotropic Regulation. *Complexity* 1(6):45-63.

Somogyi, R., Fuhrman, S., Askenazi M., Wuensche A. (1996) The Gene Expression Matrix: Towards the Extraction of Genetic Network Architectures. *Proc. Second World Congress of Nonlinear Analysts (WCNA96)*. Elsevier Science.

Thieffry, D. and Thomas R. (1998) Qualitative Analysis of Gene Networks. *Proc. Pacific Symposium on Biocomputing 1998*. In press.

Wuensche, A., Lesser, M.J. (1992) The Global Dynamics of Cellular Automata, SFI Studies in the Sciences of Complexity, Volume 1. Addison Wesley.