A Study of 3-gene Regulation Networks Using NK-Boolean Network Model and Fuzzy Logic Networking

Trina Kok and Paul Wang

Pratt School of Engineering, Duke University

1 Introduction

Boolean network theory, proposed by Stuart A. Kauffman about 3 decades ago, is more general than the cellular automata theory of von Neumann. This theory has many potential applications, and one especially significant application is in the modeling of genetic networking behavior. In order to understand the genomic regulations of a living cell, one must investigate the chaotic phenomena of some simple Boolean networks.

This chapter studies a very basic and simple 3-genes regulation network. Different combinations of the three basic logic elements: AND, OR and COMPLEMENT results in different logic functions. We study the influence of these logic functions on steady states behavior of the attractors and limit cycle patterns of cells.

In evaluating the degrees of gene expression using Boolean network theory, it is necessary to quantize the expression levels to "1" and "0". "1" indicates that the gene is expressed and a protein is formed; "0" indicates that the gene is not expressed at all. However, gene expression occurs in many stages, and it is not uncommon for the expression of a gene to cease in one of the intermediate steps. Thus, there is a need for the development of a model to represent the varying degrees of gene expression. We used Fuzzy Logic Networking to circumvent the information loss associated with quantization.

Hopefully, a complete dictionary of the classification or taxonomy, of all possible chaotic patterns can be established, as it is useful in the sense that more complex chaotic behavior resulted from gene regulation can be derived from the basic patterns in it. It is highly possible that the "reverse engineering" problem can be completely solved theoretically for the 3-gene networks.

T. Kok and P. Wang: *A Study of 3-gene Regulation Networks Using NK-Boolean Network Model and Fuzzy Logic Networking*, StudFuzz **201**, 119–151 (2006)
www.springerlink.com

2 Biological Background

Deoxyribonucleic acid (DNA) was discovered in 1869 by a Swiss biochemist, Johann Freidrich Miescher when he prepared a pure sample of nucleic acid from salmon sperm [2]. As various biologists such as Frederick Griffith [3], Hershey and Chase [4], and Oswald Avery et al [5] confirmed that DNA composes the genetic material in all cells (eukaryotic and prokaryotic), DNA has become the focus of many biological studies. Of these studies, Watson and Crick's elucidation of the DNA structure [6] remains the most important and famous. Today, DNA transcription, translation and mutation, and gene expression and regulation are enigma no more in the scientific community.

The human genome is a term used to describe the total genetic information (DNA content) in human cells, and is composed of the nuclear genome and mitochondrial genome. The nuclear genome accounts for 99.9995% of the total genetic information, the bulk of which codes for protein synthesis on cytoplasmic ribosomes, while mitochondrial genome accounts for the remaining 0.0005%. [7] The starting product of genome expression is the transcriptome, which is a term used to represent the agglomeration of ribonucleic acid (RNA) molecules synthesized from protein-coding genes [2]. The end product of genome expression is the proteome, which is the collection of proteins that subsequently contribute to the functioning of the cell [2]. In order to understand gene expression and regulation, knowledge of the processes from genome to proteome is requisite.

The gene is a part of the genome that codes for a particular protein, and composes of a sequence of nucleotides. The flow of genetic information from gene to protein is largely one-way: from DNA to RNA to protein. [7] However, the intricate details of protein synthesis are far from simple. An outline of the important reactions follows.

An enzyme, RNA polymerase, carries out transcription of RNA from DNA. In cells, DNA is packaged tightly into chromatin, which is in turn attached to various proteins that must be displaced so that RNA polymerase can contact the genes. Before transcription, the unnecessary proteins are removed and chromatin is unwound to expose the DNA. RNA polymerase and its various accessory proteins then assemble to form the transcription initiation complex. This complex binds to promoter elements on the exposed part of the DNA to signal that RNA synthesis is about to begin. As transcription begins, RNA polymerase dissociates from the transcription initiation complex and begins to catalyze the synthesis of RNA [2].

In most eukaryotic cells, the RNA transcript undergoes a series of processing reactions. This largely involves splicing and capping. During splicing, non-coding regions (introns) of RNA are removed and coding regions (exons) are ligated to create a continuous sequence of information. During capping, a nucleotide linkage is added to the 5' end of the RNA, and adenylate (AMP) residues are sequentially added to the 3' end to form a poly (A) tail. The cap and poly (A) tail serve to facilitate movement of the RNA molecule from the nucleus to the cytoplasm [7].

In the cytoplasm, information encoded on the RNA molecule is translated into proteins via the ribosomes. Ribosomes are RNA-protein complexes that help thread various amino acids in the order defined by the RNA sequence. Proteins formed from the ribosome then undergo post-translational modification where specific chemical groups are added or removed. These chemical groups tag the different proteins for different functions [7].

Gene regulation occurs at every stage of the cascade leading from genome to proteome. Proteins that make up the transcription initiation complex bind to promoter regions of the DNA to activate transcription. Depending on the nature of the binding, different amounts of RNA transcripts are produced. The proteins would thus be known as transcriptional activators. Transcriptional repressors are also found in the cell, and as the name suggests, they suppress the production of RNA transcripts. Together with post-transcriptional and post-translational regulators, transcriptional regulators coordinate the production of active protein in response to cell cycle changes and environmental stimulants.

Most of the above-mentioned regulators are proteins, products from the expression of other genes. Thus it is conceivable that the expression of one gene influences another. With the estimated 30,000 to 35,000 genes in human cells [8], the interrelation between genes sets up a convoluted network. Using location analysis and expression data, Simon et al [9] found that transcriptional activators responsible for one stage of the yeast cell cycle regulate transcriptional activators responsible for the next stage, setting up a complex regulatory network.

3 Regulatory Networks

The advent of DNA microarray technology and oligonucleotide chips [10-14] has presented much data regarding gene expression and activity profiles. The employment of gene expression data has enabled the classification of breast cancer [15], leukemias [16], and blue-cell cancers [17]. Recently, many studies have applied this data to reverse engineering. In

reverse engineering, researchers strive to evaluate a single set of regulatory interactions from samples of expression data by Schmulevich et al [18]. The ability to evaluate regulatory networks from expression data is projected to facilitate the identification of drug targets. Various models and algorithms were studied in an attempt to elucidate genetic regulatory networks, some of which include synchronous and asynchronous Boolean models [18-20], probabilistic Boolean models [18], cellular automata [21,22], Bayesian networks [23,24], Artificial Neural networks [25], (quasi) linear [26] and linear [27] networks, Petri Nets [28], Mjolsness models [29], ordinary differential equations [30], genetic programming [31], fuzzy logic [32], qualitative reasoning [33], S-systems [33], clustering [34] and yet more other approaches.

Clustering is often coupled with other algorithms and models to provide an integrative regulatory genetic network. Genes with similar expression profiles are likely to be regulated by the same processes. Clustering allows for identification of such groups of genes and further elucidation of their individual regulation. However, they fail to provide a holistic topography of the genetic regulatory network, so that various other algorithms and models have to fill the void [34].

Of all the models, Boolean models pioneered by Kauffman [35] – [36] are still the most studied. The model considers each gene symbolically as either ON/1 (expressed) or OFF/0 (not expressed), so that the continuous data obtained from microarray technology has to be quantized to these two levels. States of genes in time 't' regulate the states of genes in time 't+1' via logic functions consisting of AND, OR and COMPLEMENT connectors. As regulation proceeds among genes in a parallel manner, a synchronized genetic regulatory network evolves. An NK-Boolean network is set up by the evolution of N genes with K connectivity, where N refers to the total number of genes in the network, and K refers to the maximum number of genes that regulate some single gene. The number of possible states for such a network and the amount of data necessary for its elucidation is 2^N (Please refer to Table I). Assuming a network with maximum connectivity (K=N) like those studied by Wang et al [37-40], there are

$$\left(2^{2^N} - 2\right)^N \tag{1}$$

possible logic functions to realize a typical gene regulatory network.

Thus it may be concluded that the Boolean network model results in an uninformative discrete representation of gene expression and activity profiles, and lead to an intractable solution.

In addressing the above concerns, it was found that real regulatory networks typically have low connectivity, which translates into low K values [41]. Only a small fraction of the 2N possible gene expression states are fulfilled where unfulfilled states represent unstable states. A low K connectivity results in a tractable solution with a smaller number of possible logic functions. Working from the hypothesis that if the quantized expression and activity profiles do not provide sufficient information to separate classes of tumors, the Boolean network model is an unrealistic representation of genetic networks, Shmulevich and Zhang set out to evaluate the credibility of the Boolean network model [42]. In that study, Shmulevich and Zhang found that the Boolean network model was able to provide a clear separation between different classes of sarcomas and different subclasses of gliomas, indicating that the Boolean network model retains sufficient biological information to realistically model genetic regulatory networks [42]. Intuitively, the Boolean network model is a suitable representation of genetic networks because genetic manipulation often involves either over-expression or deletion of a gene [20].

Table 1. from [34]

Model	Data needed
Boolean, fully connected	2^N
Boolean, connectivity K	2^K (K + log(N))
Boolean, connectivity K, linearly separable	K log(N/K)
Continuous, fully connected, additive	N+1
Continuous, connectivity K, additive	K log (N/K) (*)
Pairwise correlation comparisons (clustering)	log(N)

Fully connected is where each gene can receive regulatory inputs from all other genes. Connectivity K: at most K regulatory inputs per gene. Additive, linearly separable: regulation can be modeled using a weighted sum. Pairwise correlation: significance level for pairwise comparisons based on correlation must decrease inversely proportional to number of variables. (*) conjecture.

As a Boolean network evolves in time, a sequence of states results and converges to limit cycles or attractors eventually. Information from the initial states are no longer as important and only a small number of all the possible configurations actually occur [43], composing the limit cycle or attractor.

1. *Attractor*: An attractor is a set of states, invariant under the dynamically progrssion, towards which the neighboring states in a given basin of attraction asymptotically approach in the course of dynamic evolution. An attractor is defined as the smallest unit which cannot be itself decomposed into two or more attractors with distinct basins of attraction. [44]
2. *Basin of attraction:* The set of points in the state vector space of system state variables such that initial conditions chosen in this set dynamically evolve to a particular attractor. [45]
3. *Limit cycle: A*n attracting set of state vectors to which orbits or trajectories converge and upon which trajectories are periodic. [46]
4. *Length of a limit cycle:* In the above sense, the length of a limit cycle represents its fundamental period and is equal to the number of states contained within the cycle.
5. *Basin number:* The basin number is the number of reachable states to a limit cycle or attractor.

Attractors and limit cycles of the Boolean network model can be interpreted in two ways. First, they can be seen to represent stable phenotypes of differentiated cells- muscle vs. nerve cells, or healthy vs. sick cells [35] [47]. In a non-chaotic network, Kauffman indicates that the number of attractors and limit cycles corresponds to the number of biological cell types [35]. Second, attractors and limit cycles can be regarded as cellular states-differentiation, apoptosis and cell cycle [20]. Both interpretations capture the concept of homeostasis perfectly. Homeostasis occurs when cells maintain their state despite minor disturbances in their environmental and internal stimuli. These perturbations can be interpreted as changes in state configurations of the cell, but as long as they reside within the same basin of attraction, the same attractors or limit cycles will be reached. Thus attractor or limit cycle stability increases with the size of the basin of attraction.

Regarding attractors and limit cycles as cellular states, cancer can be represented as a shift from the usually stable “differentiation” state to the “growth” state. Mutations might have reduced the size of the basin of attraction leading to the “differentiation” state, thus rendering it less stable and more susceptible to perturbations. Cancer drugs should then strive to push the cell from its “growth” state back into “differentiation” state. [20]

4 Investigation of 3-gene Boolean Network

The Boolean network model is realistic in its representation of genetic networks, capturing the essence of cell development and leading to a tractable solution. Here in this paper, we consider the Boolean network, which

are special cases of NK-networks, where each site takes on binary values of either 0 or 1, and represents gene expression states.

6. *Cellular Automata*: Cellular automata are simple mathematical idealizations of natural systems. They consist of a lattice of discrete sites, each site taking on a finite set of, say, integer values. The values of the sites evolve in discrete time steps according to deterministic rules that specify the value of each site in terms of the values of the neighboring sites. [43]

However, the cellular automata model is not a realistic model for biological natural systems.

In this study, we attempt to evaluate the evolution pattern of a NK-Boolean network whose evolution depends only on its two neighboring sites and itself. In addition, we assume N (total number of genes in the network) =3 and K (connectivity) =3, so that there are $2^N = 8$ possible states and

$$\left(2^{2^N} - 2\right)^N = 16{,}387{,}064 \tag{2}$$

possible combinations of logic function. As we believe that networks with larger N's may be broken down into networks of N = 2 or 3, we studied NK-Boolean network of the 3-gene network. (A study on the 2-gene network can be found in [38]). Of the 16,387,064 possible logic functions, about 150 examples were evaluated by hand, resulting in diagrams similar to Figure 1 and 2. A', B', C' represent genes at time 't+1' and A, B, C, represent genes at time 't'. Figure 1 illustrates two limit cycles, one with length 2 (L2) and the other with length 6 (L6). Both limit cycles have a basin number of 0. Figure 2 illustrates two attractors, one with basin number 0 (B0) and the other with basin number 6 (B6). In the syntax of this study, logic functions that involve no logic connectors are termed "PLAIN". Conversely, logic functions that involve AND, OR, COMPLEMENT connectors are termed "AND", "OR" and "NOT" respectively.

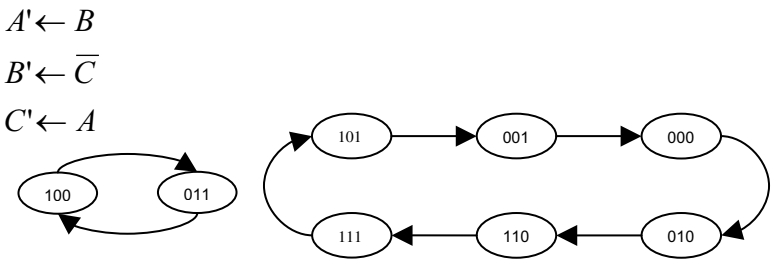


Fig. 1. An example of a 3-gene network with two limit cycles

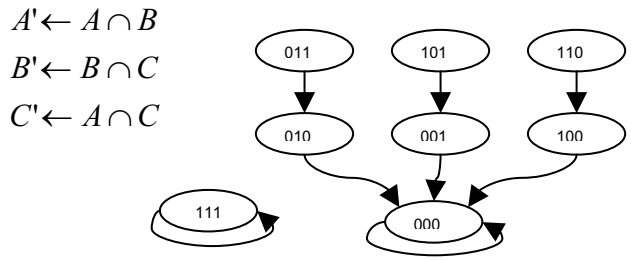


Fig. 2. An example of a 3-gene network with two attractors

5 Evolutionary Patterns

We evaluated and grouped the evolution patterns resulting from different types of logic functions. Several general rules that govern the evolution of NK-Boolean network patterns were discovered. To better summarize and compare these NK-Boolean network patterns, the logic functions and their corresponding evolution patterns are grouped into tables as follow. Figures are attached in a separate appendix for easy reference. A comprehensive dictionary is also appended as a classification of the logic functions and the evolution patterns.

Since “Reverse Engineering” has been identified by biologists as an extremely important problem, we chose to explore the possibility of solving this problem via a dictionary approach. Because it may be possible to develop some effective algorithms via heuristic arguments, having some insight about this issue is very important, even if currently far from solving the real biological problem. Many algorithms have been discovered based on biologically inspired problems.

Table 2. NK-Boolean Network patterns of PLAIN Functions

Logic Function	Logic Description	Network Description	Fig
A' ← A B' ← B C' ← C	3 self regulations	8 B0 attractors	i
A' ← B B' ← C C' ← A	No self regulation No repeated regulation	2 L3, B0 limit cycles 2 B0 attractors	ii
A' ← B B' ← A C' ← C	1 self regulation No repeated regulation	2 L2, B0 limit cycles 4 B0 attractors	iii
A' ← B B' ← A C' ← A	No self regulation 1 repeated regulation	1 L2, B2 limit cycles 2 B1 attractors	iv
A' ← A B' ← A C' ← B	1 self regulation and it is repeated	0 limit cycle 2 B3 attractors	v
A' ← A B' ← B C' ← A	2 self regulation 1 repeated regulation	0 limit cycle 4 B1 attractors	vi
A' ← A B' ← A C' ← A	1 self regulation 2 repeated regulation	0 limit cycle 2 B3 attractors	vii

Logic functions represent one example only.

Observations of PLAIN functions:

1. Attractors always include "000" and "111".

Table 3. NK-Boolean Network Patterns of NOT Functions

Logic Function	Logic Description	Network Description	Fig
A' ← $\overline{B}$ B' ← C C' ← A	1 NOT No self regulation No repeated regulation	1 L6, B0 limit cycle 1 L2, B0 limit cycle 0 attractor	viii
A' ← $\overline{C}$ B' ← B C' ← A	1 NOT 1 self regulation ≠ NOT No repeated regulation	2 L4, B0 limit cycles 0 attractor	ix

A' ← $\overline{A}$ B' ← C C' ← B	1 NOT 1 self regulation = NOT No repeated regulation	4 L2, B0 limit cycles 0 attractor	x
A' ← $\overline{C}$ B' ← $\overline{A}$ C' ← B	2 NOTs No self regulation No repeated regulation	2 L3, B0 limit cycles 2 B0 attractors	ii
A' ← A B' ← $\overline{C}$ C' ← $\overline{B}$	2 NOTs 1 self regulation ≠ NOT No repeated regulation	2 L2, B0 limit cycles 4 B0 attractors	iii
A' ← $\overline{B}$ B' ← A C' ← $\overline{C}$	2 NOTs 1 self regulation = NOT No repeated regulation	2 L4, B0 limit cycles 0 attractor	ix
A' ← $\overline{C}$ B' ← $\overline{A}$ C' ← $\overline{B}$	3 NOTs No self regulation No repeated regulation	1 L6, B0 limit cycle 1 L2, B0 limit cycle 0 attractor	viii
A' ← $\overline{C}$ B' ← $\overline{B}$ C' ← $\overline{A}$	3 NOTs 1 self regulation = NOT No repeated regulation	4 L2, B0 limit cycles 0 attractor	x
A' ← f(A) B' ← f(B) C' ← f(C)	1/2/3 NOTs 3 self regulation No repeated regulation	4 L2, B0 limit cycles 0 attractor	x
A' ← $\overline{A}$ B' ← $\overline{A}$ C' ← $\overline{C}$	1 repeated regulation, regardless self or not	Basin number = Length	eg. iv

Logic functions represent one example only.

Observations of NOT functions:

1. NOT functions always lead to at least 1 limit cycle.
2. If there is repeated regulation, a NK-Boolean network pattern will result where the basin number is equal to the length of the limit cycle.

Table 4. NK-Boolean Network Patterns of AND Function

Logic Function	Logic Description	Network Description	Fig
A' ← A ∩ B B' ← B ∩ C C' ← A ∩ C	3 (AND 2)s No repeated regulation	1 B6 attractor 1 B0 attractor	xi
A' ← A ∩ C B' ← B ∩ C C' ← B ∩ C	3 (AND 2)s 1 or 2 repeated regulation	No. of branches leading to attractor =3 + No. of repeated regulation	eg. xii
A' ←A ∩ B ∩ C B' ← A C' ← B	1 (AND 3)s	No general pattern observed	eg. xiv
A' ← A ∩ B ∩ C B' ← C C' ← A ∩ B ∩ C	2 (AND 3)s un-AND gene is not self regulated	1 B6 attractor 1 B0 attractor	xvi
A' ← A ∩ B ∩ C B' ← B C' ← A ∩ B ∩ C	2 (AND 3)s un-AND gene is self regulated	1 B3 attractor 1 B2 attractor 1 B0 attractor	xvii
A' ← A ∩ B ∩ C B' ← A ∩ B ∩ C C' ← A ∩ B ∩ C	3 (AND 3)s	1 B6 attractor 1 B0 attractor	xviii

Logic functions represent one example only

Observations of AND functions

1. AND functions always lead to attractors of "000" and "111"
2. "000" is always the dominant attractor with a greater basin number, while "111" is the least dominant attractor with the smallest basin number

Table 5. NK-Boolean Network Patterns of OR Functions

Logic Function	Logic Description	Network Description	Fig
A' ← A ∪ B B' ← B ∪ C C' ← A ∪ C	3 (OR 2)s No repeated regulation	1 B6 attractor 1 B0 attractor	xi
A' ← A ∪ B B' ← A ∪ B C' ← A ∪ C	3 (OR 2)s 1 or 2 repeated regulation	No. of branches leading to attractor = 3+ No. of repeated regulation	eg. xix

A’ ← A ∪ B ∪ C B’ ← A C’ ← B	1 (OR 3)s	No general pattern observed	eg. xx
A’ ← A ∪ B ∪ C B’ ← C C’ ← A ∪ B ∪ C	2 (OR 3)s un-OR gene is not self regulated	1 B6 attractor 1 B0 attractor	xvi
A’ ← A ∪ B ∪ C B’ ← B C’ ← A ∪ B ∪ C	2 (OR 3)s un-OR gene is self regulated	1 B3 attractor 1 B2 attractor 1 B0 attractor	xvii
A’ ← A ∪ B ∪ C B’ ← A ∪ B ∪ C C’ ← A ∪ B ∪ C	3 (OR 3)s	1 B6 attractor 1 B0 attractor	xviii

Logic Functions represent one example only.

Observations of OR functions

1. OR functions always lead to attractors of “000” and “111”
2. “111” is always the dominant attractor with a greater basin number, while “000” is the least dominant attractor with the smallest basin number

More complicated functions that involve combination of logic connectors like those shown in Figure 3 and 4 were also evaluated, but no general evolution pattern was discovered for any of these functions.

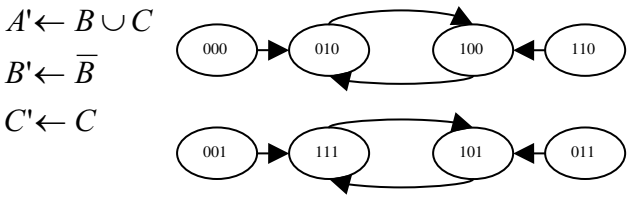


Fig. 3. An example of a complicated 3-gene network

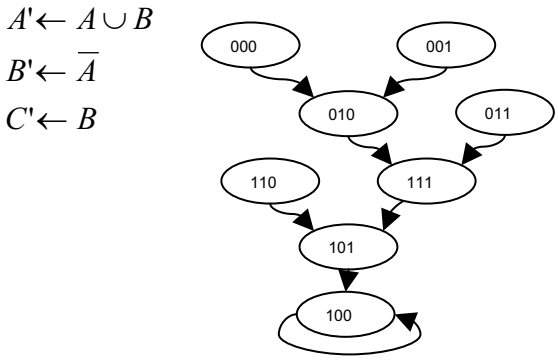


Fig. 4. An example of a complicated 3-gene network

6. Results

The concept of a "dictionary-like" solution to the "reverse-engineering" problem of a (N, K) = (3, 3) Boolean network yields a total possible combinations of logic functions of $({}_2 2^N - {}_2)^N = ({}_2 2^3 - {}_2)^3 = 16{,}387{,}064$ seem hopelessly complex. The theoretical solution, to say the least, is insurmountable complex. On the optimistic side, the situation is much better than one would think. Based upon the results presented in this paper, we have developed many heuristic rules via induction-deduction methods. Furthermore, the objective of a dictionary-like solution is attractive and feasible because the problem itself is tractable in some way if one sets up the problem as a patterns classification problem.

To begin with, one may designate a pattern vector space with 16,387,064 vector elements. In this pattern space, we may define the following features in the feature vectors space.

- $x^a_i = 1,2, \ldots\ldots, 8$, with i = 8 as maximum = number of attractors
- $x^a_i(B(i)) = x^a_1(B(1)), x^a_2(B(2)), \ldots \ldots, x^a_8(B(8))$, with $x^a_8(B(8)) = 0$ as limit = number of attractors basin
- $x^l_j = 1,2,3,4,$ with j = 4 as maximum = number of limit cycles
- $x^l_j(B(j)) = x^l_1(B(1)), x^l_2(B(2)), x^l_3(B(3)), x^l_4(B(4))$, with $x^l_4(B(4)) = 0$ as limit
- $x^l_j(L(j)) = x^l_1(L(1)), x^l_2(L(2)), x^l_3(L(3)), x^l_4(L(4))$, with $x^l_4(L(4)) = 2$ as limit = the length of limit cycle

To illustrate an example, say for the case of Figure (xii).

- $x^{a}_{3} = 3$
- $x^{a}_{1}(B(1)) = 5$,
- $x^{a}_{2}(B(2)) = 0$,
- $x^{a}_{3}(B(3)) = 0$,
- $x^{l}_{1} = 0$,
- $x^{l}_{1}(L(1)) = 0$

With the above specific feature vector, the significant question to be answered now would be, what is the logic functions for this output observations?

It is more interesting to observe that strictly speaking, this dictionary-like solution really is not a pattern recognition problem. Usually, the reverse problem for pattern recognition problem is not one-to-one (or unique inverse mapping). This approach nevertheless is very interesting in that the problem resembles the mathematical linguistic pattern recognition problem.

Close observation of the above tables leads to a number of general rules that govern the evolution of NK-Boolean network patterns. These links between logic functions and NK-Boolean network evolution patterns will aid efforts in "reverse engineering". We list our observations as below

- Rule 1: NOT functions produce at least 1 limit cycle. Conversely, pure AND and OR functions only produce attractors
- Rule 2: For NOT functions, the below function is a special case where 4 limit cycles with length 2 always result.
  A' ← f(A)
  B' ← f(B)
  C' ← f(C)
- Rule 3: For NOT functions, the presence of repeated regulation will result in a NK-Boolean network pattern with length equal to its basin number.
- Rule 4: For AND and OR functions, attractors of "000" and "111" are always present.
- Rule 5: For AND functions, the dominant attractor is "000", and the least dominant attractor is "111".
- Rule 6: For OR functions, the dominant attractor is "111", and the least dominant attractor is "000".
- Rule 7: For 3 (AND 2) functions such as the below, the number of branches leading to the attractor is 3+number of repetitions. Therefore,

the range of branches leading to the attractors is between 3 and 5 (since there can be at most 2 repetitions)

$A' \leftarrow A \cap B$
$B' \leftarrow B \cap C$
$C' \leftarrow A \cap C$

In summary, we view the results of this study fructified, insightful and satisfactory.

7 Problems of the Boolean Network

Whilst the Boolean network provides a suitable and tractable solution in modeling gene regulatory networks, it fails to address several inherent characteristics of real biological genetic networks.

First, the Boolean network is deterministic and real genetic regulatory networks are stochastic [48]. A deterministic network consists of gene expression states with only one output, but a stochastic network consists of gene expression states with two or more outputs. With the same inputs and the same initial gene expression state, a stochastic network may produce a different output at one time and another output at a later time. This problem could be addressed by probabilistic Boolean networks that take into account the probability of each output occurrence [18].

Second, the Boolean network does not accommodate noise introduced by microarray and measurement techniques [18]. However, this property of real genetic regulatory networks is considered in the algorithm by Akutsu et al [33] in their evaluation of Boolean genetic networks with noise.

Third, the absolute binary values of Boolean network does not account for the varying degrees at which regulators affect gene expression. By placing weights on certain regulators, neural networks attempt to model the real genetic regulatory networks more realistically [25].

Fourth, negative feedback with a moderate feedback gain is often used in real genetic regulatory networks to stabilize the system, but negative feedbacks in Boolean networks only serve to destabilize the system [34].

Fifth, the Boolean network operates synchronously as each gene expression state gets updated in time-steps, but real genetic networks operate asynchronously.

8 Investigation of Fuzzy Logic Networking

In addressing the problem of representing varying degrees of gene expression, we study Fuzzy Logic Networking as a tool. Here, $\boldsymbol{X}$ represents the set of expressed genes and μ_x represents a function that maps elements from the universal set $\boldsymbol{U}$ to $\boldsymbol{X}$. $\boldsymbol{X}$ connotes the idea of "expressed" genes, and μ_x corresponds to the degree of expression of each gene. μ_x is known as the membership function and X(u) is degree of membership of u in the fuzzy subset of X [49].

Table 6. Five Fuzzy Logics from [50]

		x or y	x and y
Logic 1	CFMQVS	min(1,x+y)	x*y
Logic 2	max/min	max(x, y)	min(x,y)
Logic 3	probabilistic	x+y-x*y	x*y
Logic 4	MV	min(1,x+y)	max(0,x+y-1)
Logic 5	gcd/lcm	gcd(x,y)	lcm(x,y)

In the case of fuzzy subsets, we employ the same operations used in the Boolean representation i.e. AND, OR and COMPLEMENT. Due to the range of membership values, there is no one way of carrying out the AND, OR and COMPLEMENT operations on the fuzzy subsets. Table 6 presents five logics that were discussed by Reiter [50]. As observed by Reiter, each of the logic has different characteristics that are worth noting. Logic 2 is a common operation based on maximum and minimum replacing, and it used in both [49] and [51]. Reiter observed that Logic 1 has a fairly high-valued "OR" and a fairly low-valued "AND"; Logic 3 has the same "AND" as Logic 1 but the "OR" values are sometimes lower; Logic 4 has the same "OR" as Logic 1 but there are more 0 values present in "AND". Based on the operation on two fuzzy subsets X and Y, we generated figures that showed the differences amongst these. The plots compare only Logic 1 to Logic 4 as Logic 5 does not map back to the interval of [0,1]. Figure 5 shows that Logic 2 gives higher "AND" values than Logic 3, and Logic 3 in turn gives higher "AND" values than Logic 1 and 4. Figure 6 shows that Logic 4 gives higher "OR" values than Logic 1 and 3, and Logic 1 and 3 gives higher "OR" values than Logic 2. From the plots, the logic functions are approximately similar.

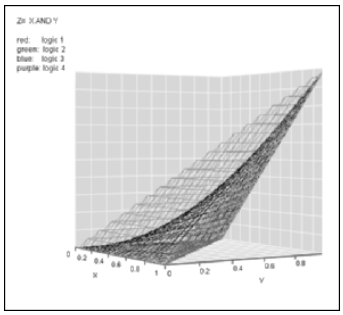


Fig. 5. Comparison of "AND" values. Logic 1 is same as logic 3.

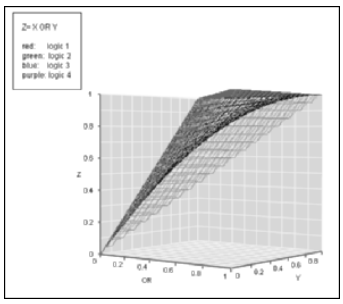


Fig. 6. Comparison of "OR" values. Logic 1 is same as logic 4.

9 Fuzzy Study of Regulation Networks

A preliminary study of 3-gene regulation networks using fuzzy sets was carried out. About 150 examples were evaluated by hand. We observed that different (i) logic operations (Logic1, 2, 3 or 4), (ii) logic functions, and (iii) initial membership values led to different attractor and limit cycles for the 3-gene regulation network. Compared to the study of 3-gene regulation networks using NK-Boolean network, (i) and (iii) are additional parameters in the evolution pattern of the 3-gene network.

Evolution patterns of 3-gene networks governed by PLAIN or NOT functions are unaffected by parameter (i). Assuming different membership values at time 't_0', 3-gene regulation networks governed by PLAIN or NOT functions generally evolve to limit cycles, as long as self-regulation does not take place. 3-gene regulation networks governed by AND or OR

functions generally evolve to attractors. We note the similarity between fuzzy analysis and NK-Boolean analysis of 3-gene regulation networks.

In order to meaningfully evaluate the attractor and limit cycle patterns, we made two assumptions. 1.) A, B, C have different initial membership values and 2.) Logic 2 is used to the AND and OR operations. Assumption 1 is intuitively reasonable as it is unlikely that two genes will have exactly the same extent of expression.

We grouped the logic functions and their corresponding evolution patterns into tables as below and observed a number of general rules that govern the evolution of Fuzzy Logic network patterns. We list our observations below after Table 7-10.

Table 7. Fuzzy Logic Network Patterns of PLAIN Functions

Logic Function	Logic Description	Network Description	Reason
A' ← A B' ← B C' ← C	3 self regulations	B0 attractors	
A' ← B B' ← C C' ← A	No self regulation No repeated regulation	L3 limit cycle	Exchange of results- requires 3 steps to repeat
A' ← B B' ← A C' ← C	1 self regulation No repeated regulation	L2 limit cycle	Toggling of values between 2 non-self regulated gene
A' ← B B' ← A C' ← A	No self regulation 1 repeated regulation	L2 limit cycle	Toggling between 2 sets of values
A' ← A B' ← A C' ← B	1 self regulation and it is repeated	B2 attractors to value of repeated gene	Requires 2 steps before all follows repeated gene
A' ← A B' ← B C' ← A	2 self regulation 1 repeated regulation	B1 attractor	Since 2 genes are self-regulated, requires only step to force non-self regulated gene
A' ← A B' ← A C' ← A	1 self regulation 2 repeated regulation	B1 attractor	

Table 8. Fuzzy Logic Network Patterns of NOT functions

Logic Function	Logic Description	Network Description	Reason
A' ← $\overline{B}$ B' ← C C' ← A	1 NOT No self regulation No repeated regulation	L6 limit cycle	
A' ← $\overline{C}$ B' ← B C' ← A	1 NOT 1 self regulation ≠ NOT No repeated regulation	L4 limit cycle	NOT adds double the steps
A' ← $\overline{A}$ B' ← C C' ← B	1 NOT 1 self regulation = NOT No repeated regulation	L2 limit cycle	
A' ← $\overline{C}$ B' ← $\overline{A}$ C' ← B	2 NOTs No self regulation No repeated regulation	L3 limit cycle	
A' ← A B' ← $\overline{C}$ C' ← $\overline{B}$	2 NOTs 1 self regulation ≠ NOT No repeated regulation	L2 limit cycle	Two non-self regulated genes are negated, takes two steps to revert back
A' ← $\overline{B}$ B' ← A C' ← $\overline{C}$	2 NOTs 1 self regulation = NOT No repeated regulation	L4 limit cycle	
A' ← $\overline{C}$ B' ← $\overline{B}$ C' ← $\overline{A}$	3 NOTs No self regulation No repeated regulation	L6 limit cycle	NOT adds double the steps
A' ← f(A) B' ← f(B) C' ← f(C)	1/2/3 NOTs 3 self regulation No repeated regulation	L2 limit cycle	Self regulation and NOT ensures only a toggle of 2 values

Table 9. Fuzzy Logic Network patterns of AND Functions

Logic Function	Logic Description	Network Description	Reason
A' ← A ∩ B B' ← B ∩ C C' ← A ∩ C	3 (AND 2)s No repeated regulation	B2 attractor	Need two steps to pick the minimum of 3 genes
A' ← C B' ← A ∩ B C' ← B ∩ C	un-AND gene is not self regulated	B3 attractor	
A' ← A ∩ B B' ← B C' ← B ∩ C	2 (AND 2)s un-AND gene is self regulated	If un-AND gene holds highest value: B0 attractor If un-AND gene holds middle value: B1 attractor If un-AND gene holds middle value: B2 attractor	
A' ← A ∩ B ∩ C B' ← C C' ← A ∩ B ∩ C	2 (AND 3)s un-AND gene is not self regulated	B2 attractor	Since un-AND gene is not self regulated, require 2 steps to achieve minimum of 3
A' ← A ∩ B ∩ C B' ← B C' ← A ∩ B ∩ C	2 (AND 3)s un-AND gene is self regulated	B1 attractor	Since un-AND gene is self regulated, require 1 step to achieve minimum of 3
A' ← A ∩ B ∩ C B' ← A ∩ B ∩ C C' ← A ∩ B ∩ C	3 (AND 3)s	B1 attractor	

Table 10. Fuzzy Logic Network Patterns of OR Functions

Logic Function	Logic Description	Network Description	Reason
A' ← A ∪ B B' ← B ∪ C C' ← A ∪ C	3 (OR 2)s No repeated regulation	B2 attractor	Need two steps to pick the maximum of 3 genes
A' ← C B' ← A ∪ B C'← B ∪ C	un-OR gene is not self regulated	B3 attractor	
A' ← A ∪ B B' ← B C' ← B ∪ C	2 (OR 2)s un-OR gene is self regulated	If un-OR gene holds highest value: B2 attractor If un-OR gene holds middle value: B1 attractor If un-OR gene holds middle value: B0 attractor	
A' ← A ∪ B ∪ C B' ← C C' ← A ∪ B ∪ C	2 (OR 3)s un-OR gene is not self regulated	B2 attractor	Since un-OR gene is not self regulated, require 2 steps to achieve maximum of 3
A' ← A ∪ B ∪ C B' ← B C' ← A ∪ B ∪ C	2 (OR 3)s un-OR gene is self regulated	B1 attractor	Since un-OR gene is self regulated, require 1 step to achieve maximum of 3
A' ← A ∪ B ∪ C B' ← A ∪ B ∪ C C' ← A ∪ B ∪ C	3 (OR 3)s	B1 attractor	

We observed the following general rules

- Rule 1: NOT functions produce limit cycles.
- Rule 2:

Table 11. Table for Rule 2

Example	Logic Description	Rules
A' ← A ∩ B B' ← B C' ← B ∩ C	2 (AND 2)s un-AND gene is self regulated	If un-AND gene holds highest value: B0 attractor If un-AND gene holds middle value: B1 attractor If un-AND gene holds middle value: B2 attractor

- Rule 3:

Table 12. Table for Rule 3

Example	Logic Description	Rules
A' ← A ∪ B B' ← B C' ← B ∪ C	2 (OR 2)s un-OR gene is self regulated	If un-OR gene holds highest value: B2 attractor If un-OR gene holds middle value: B1 attractor If un-OR gene holds middle value: B0 attractor

- Rule 4:

Table 13. Table for Rule 4

Example	Logic Description	Rules
A' ← A ∩ B B' ← B ∩ C C' ← A ∩ C	3 (AND 2)s No repeated regulation	B2 attractor results as two steps are needed to pick the minimum of 3 genes

- Rule 5:

Table 14. Table for Rule 5

Example	Logic Description	Rules
A' ← A ∪ B B' ← B ∪ C C' ← A ∪ C	3 (OR 2)s No repeated regulation	B2 attractor as two steps are needed to pick the maximum of 3 genes

10 Conclusion

As the cell ages and undergoes irreversible changes, it either goes into an invariant state or progresses through a periodic cycle. The irreversible evolution of the NK-Boolean network and Fuzzy Logic network mimics such changes very well. Future work would build on the basic connectors studied in this paper, and include computer simulations of more complicated functions involving combinations of logic connectors.

The most significant result obtained from this study is a fundamental understanding of the stochastic behavior of a cell with simple assumption of a simple three-gene network. It is affirmative that a "reverse-engineering" problem can be solved, at least theoretically via induction and reduction approach. In general, the realistic biological behavior is so complicated that it can be seen to be at the edge-of-the-chaotics. Usually, a theoretical and mathematical derivation is very difficult. Nevertheless, a simplified analysis based upon a simple model would definitely provide a much needed visualization of the biological behavior and associated phenomenon. It is only through mathematical analysis would there be a better chance in understanding the complex phenomenon. This is partially true when the simplified model can be viewed as the basic building blocks of a complicated situation. Only then would there be some hope for some progress.

Appendix of Figures

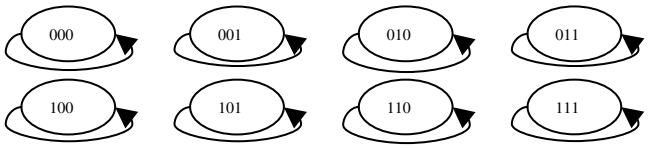


Fig. (i).

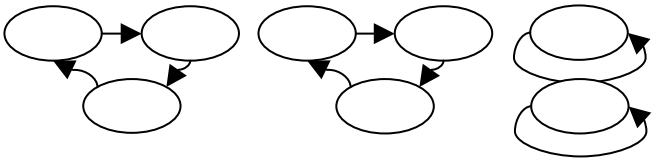


Fig. (ii).

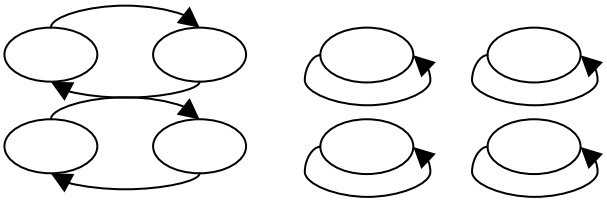


Fig. (iii).

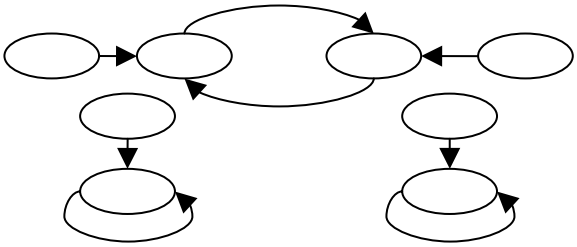


Fig. (iv).

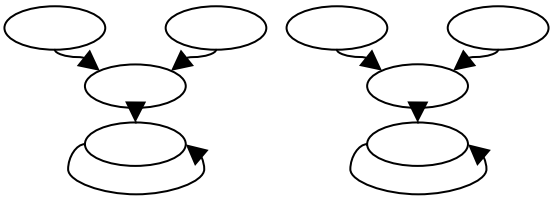


Fig. (v).

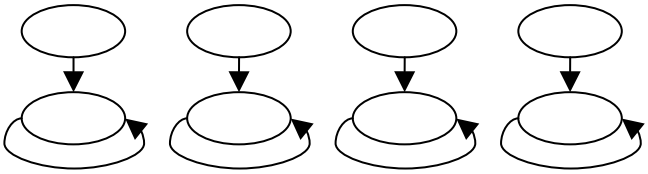


Fig. (vi).

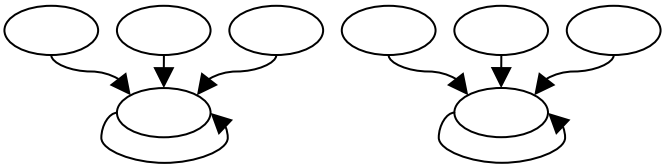


Fig. (vii).

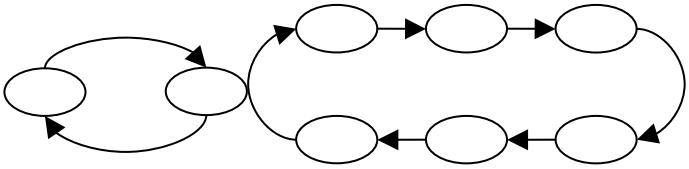


Fig. (viii).

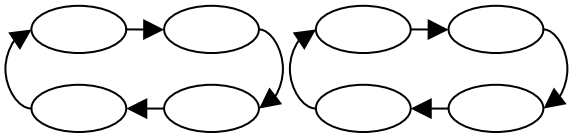


Fig. (ix).

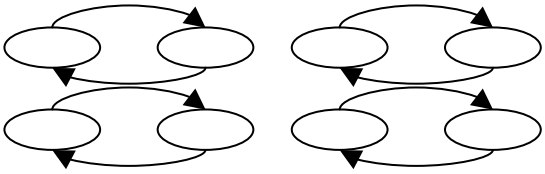


Fig. (x).

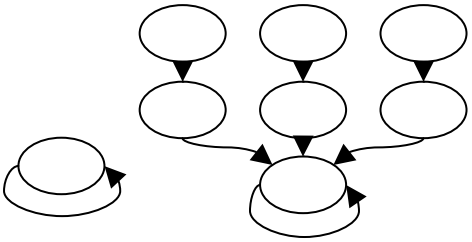


Fig. (xi)

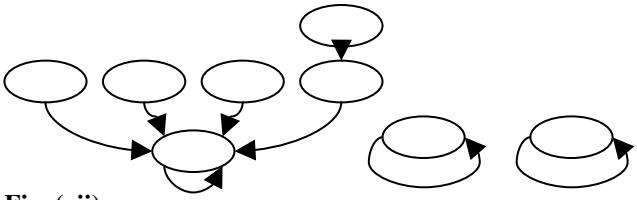


Fig. (xii).

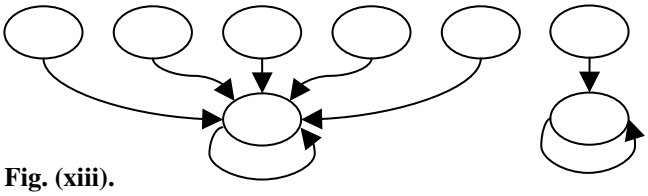


Fig. (xiii).

Appendix of Dictionary

Pure Limit Cycles

– 1 limit cycle:
– 2 limit cycle:
(a) 1 NOT (1 self regulation ≠ NOT, no repeats) = L4/L4
eg.

$A' \leftarrow \overline{C}$
$B' \leftarrow B$
$C' \leftarrow A$

(b) 1 NOT (no self regulation, no repeats) = L6/L2
eg.
$A' \leftarrow \overline{B}$
$B' \leftarrow C$
$C' \leftarrow A$

(c) 2 NOTs (1 self regulation = NOT, no repeats) = L4/L4
eg.
$A' \leftarrow \overline{B}$
$B' \leftarrow A$
$C' \leftarrow \overline{C}$

(d) 3 NOTS (no self regulation, no repeats) = L6/L2
eg.
$A' \leftarrow \overline{C}$
$B' \leftarrow \overline{A}$
$C' \leftarrow \overline{B}$

– 3 limit cycles:
– 4 limit cycles:
(a) 1 NOT (1 self regulation = NOT, no repeats) = L2/L2/L2/L2
eg.
$A' \leftarrow \overline{A}$
$B' \leftarrow C$
$C' \leftarrow B$

(b) 3 NOTs (1 self regulation = NOT, no repeats) = L2/L2/L2/L2
eg.
$A' \leftarrow f(A)$
$B' \leftarrow f(B)$
$C' \leftarrow f(C)$

Pure Attractors

– 1 attractor:
– 2 attractor:
(a) 2(AND 3) (un-AND gene is not self-regulated) = B6/B0

eg.

$A' \leftarrow A \cap B \cap C$
$B' \leftarrow C$
$C' \leftarrow A \cap B \cap C$

(b) 3(AND 3) = B6/B0

eg.

$A' \leftarrow A \cap B \cap C$
$B' \leftarrow A \cap B \cap C$
$C' \leftarrow A \cap B \cap C$

(c) 2(OR 3) (un-OR gene is not self regulated) = B6/B0

eg.

$A' \leftarrow A \cup B \cup C$
$B' \leftarrow C$
$C' \leftarrow A \cup B \cup C$

(d) 3(OR 3) = B6/B0

eg.

$A' \leftarrow A \cup B \cup C$
$B' \leftarrow A \cup B \cup C$
$C' \leftarrow A \cup B \cup C$

– 3 attractor:

(a) 2(AND 3)(un-AND gene is self regulated)= B3/B2/B0

eg.

$A' \leftarrow A \cap B \cap C$
$B' \leftarrow B$
$C' \leftarrow A \cap B \cap C$

(b) 2(OR 3) (un-OR gene is self regulated) = B3/B2/B0

eg.

$A' \leftarrow A \cup B \cup C$
$B' \leftarrow B$
$C' \leftarrow A \cup B \cup C$

– 4 attractor:

(a) Plain (2 self regulation, 1 repeat) = B1/B1/B1/B1

$A' \leftarrow A \quad A \quad A \quad B \quad A \quad C$
$B' \leftarrow B \text{ or } A \text{ or } B \text{ or } B \text{ or } C \text{ or } B$
$C' \leftarrow A \quad C \quad B \quad C \quad C \quad C$

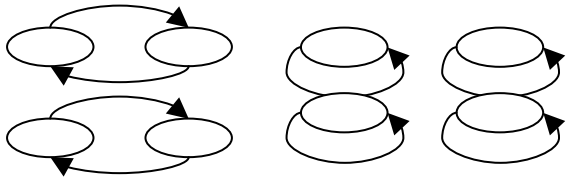
– 5 attractor:

– 6 attractor:

– 7 attractor:
– 8 attractor:
  (a) Plain(3 self regulations) = B0/B0/B0/B0/B0/B0/B0/B0
  A' ← A
  B' ← B
  C' ← C

Mixture

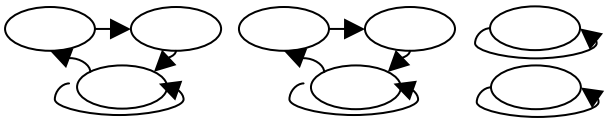
– 2 limit cycles and 4 attractors



  (a) Plain (1 self regulation, no repeats) = 4 B0 attractors, 2 L2 limit cycles
  A' ← B A C
  B' ← A or C or B
  C' ← C B A

  (b) 2 NOTs (1 self regulation ≠ NOT, no repeats) = 4 B0 attractors, 2 L2 limit cycles
  eg.
  A' ← A
  B' ← $\overline{C}$
  C' ← $\overline{B}$

– 2 limit cycles and 2 attractors



  (a) Plain (no self regulation, no repeats) = 2 B0 attractors, 2 L3 limit cycles
  A' ← B C
  B' ← C or A
  C' ← A B

(b) 2 NOTs (no self regulation, no repeats) = 2 B0 attractors, 2 L3 limit cycles

eg.

$A' \leftarrow C^c$

$B' \leftarrow \overline{A}$

$C' \leftarrow B$

References

1. Brown TA (2002) Genomes 2. John Wiley & Sons Inc, 1-69 and 219-347
2. Alberts B, Johnson A, Lewis J, Raff M, Roberts K, Walter P (2002) The Cell. Garland Science, 375-394
3. Griffith F (1928) The significance of pneumococcal types. J Hygiene 27:113-159
4. Hershey AD, Chase M (1952) Independent functions of viral protein and nucleic acid in growth of bacteriophage. J Gen Physiol 36:39-56
5. Avery OT, MacLeod CM, McCarty M (1944) Studies on the chemical nature of the substance inducing transformation of pneumococcal types. J Exp Med 79:137-158
6. Watson JD, Crick FHC (1953) Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid. Nature 171:737-738
7. Strachan T, Read A (1999) Human Molecular Genetics 2. Wiley-Liss, Chapter 7
8. Human Genome Management Information System (2003) Quick Facts, http://www.ornl.gov/sci/techresources/Human_Genome/resource/media.shtml
9. Simon I, Barnett J, Hannett N, Harbison CT, Rinaldi NJ, Volkert TL, Wyrick JJ, Zeitliner J, Gifford DK, Jaakola TS, Young RA (2001) Serial regulation of transcriptional regulators in the yeast cell cycle. Cell 106:697-708
10. Schena M, Shalon D, Davis RW, Brown PO (1995) Quantitative monitoring of gene expression patterns with a complementary DNA microarray. Science 270:467-470
11. Celis JE, Kruhoffer M, Gromova I, Frederiksen C, Ostergaard M, Thyjaer T, Gromov P, Yu J, Palsdottir H, Magnusson N, Orntoft TF (2000) Gene expression profiling: Monitoring transcription and translation products using DNA microarrays and proteomics. FEBS Letters 480:1:2-16
12. Hughes TR, Mao M, Jones AR, Burchard J, Marton MJ, Shannon KW, Lefkowitz SM, Ziman M, Schelter JM, Meyer MR, Kobayshi S, Davis C, Dai H, He YD, Stephaniants SB, Cavet G, Walker WL, West A, Coffey E, Shoemaker DD, Stoughton R, Blanchard AP, Friend SH, Linsley PS (2001) Expression profiling using microarrays fabricated by an ink-jet oligonucleotide synthesizer. Nature Biotechnol 19:342-347
13. Lipshutz RJ, Fodor SPA, Gingeras TR, Lockart DJ (1999) High density synthetic oligonucletide arrays. Nature genetics 21:20-49

14. Lockhart DJ, Winzeler, EA (2000) Genomics, gene expression and DNA arrays. Nature 405:827-836
15. Hedenfalk I, Duggan D, Chen Y, Radmacher M, Bittner M, Simon R, Meltzer P, Gusterson B, Estellar M, Raffeld M, Yakhini Z, Ben-Dor A, Dougherty E, Kononen J, Bubendorf L, Fehrle W, Pittaluga S, Gruverger S, Loman N, Johannsson O, Olsson H, Wifond B, Sauter G, Kallioniemi OP, Borg A, Trent J (2001) Gene expression profiles distinguish hereditary breast cancers. New England J Med 34:539-548
16. Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coler H, Loh ML, Downing JR, Caligiuri MA, Bloomfield CD, Lander ES (1999) Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286:531-537
17. Khan J, Wei JS, Ringner M, Saal LH, Ladanyi M, Westermann F, Berthold F, Schwab M, Antonescu CR, Peterson C, Meltzer PS (2001) Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nature Medicine 7:673-679
18. Shmulevich I, Dougherty ER, Zhang W (2002) From Boolean to probabilistic Boolean networks as models of genetic regulatory networks. Proceedings of the IEEE 90:11:1778-1792
19. Liang S, Fuhrman S, Somogyi R (1998) REVEAL, A general reverse engineering algorithm for inference of genetic network architectures. Pacific Symposium on Biocomputing 3:18-29
20. Huang S (1999) Gene expression profiling, genetic networks, and cellular state: an integrating concept for tumorigenesis and drug discovery. J Mol Med 77:469-480
21. de Sales JA, Martins ML, Stariolo DA (1997) Cellular automata model for genetic networks. Physical Review E 55:3:3262-327
22. Ermentrout GB, Edelstein-Keshet L (1993) Cellular automata approaches to biological modeling. J theor Biol 160:97-133
23. Friedman N, Linial M, Nachman I, Pe'er D (2000) Using Bayesian networks to analyze expression data. J Computation Biology 7:601-620
24. Friedman N, Goldszmidt M, Wyner A (1999) Data analysis with Bayesian networks: A bootstrap approach. Proc Fifteenth Conf On Uncertainty in Artificial Intelligence (UAI)
25. Weaver D, Workman C, Stormo G (1999) Modeling regulatory networks with matrices. Pacific Symposium on Biocomputing 4:112-123
26. D'Haeseleer P, Wen X, Fuhrman S, Somogyi R (1999) Linear modeling of mRNA expression levels during cns development and injury. Pacific Symposium on Biocomputing 4:29-40
27. van Someren EP, Wessels LFA, Reinders MJT (2000) Linear modeling of genetic networks from experimental data. Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology 355-366
28. Matsuno H, Doi A, Nagasaki M, Miyano S (2000) Hybrid Petri Net representation of gene regulatory network. Proc Pacific Symposium on Biocomputing 5:341-352

29. Mjolsness EDS, Reinitz J (1991) A connectionist model of development. J of Theoretical Biology 152:429-453
30. Chen T, He H and Church G (1999) Modeling gene expression with differential equations. Pacific Symposium on Biocomputing 4:29-40
31. Koza J, Mydlowec W, Lanza G, Yu J, Keane MA (2001) Reverse engineering of metabolic pathways from observed data using genetic programming. Pacific Symposium on Biocomputing 434-445
32. Thieffry D, Thomas R (1995) Dynamical behavior of biological regulatory networks-II Immunity control in bacteriophage lambda. Bull Math Biol, 57: 277-297
33. Akutsu T, Miyano S, Kuhara S (2000) Algorithms for inferring qualitative models of biological networks. Pacific Symposium on Biocomputing 5:290-301
34. D'Haeseleer P, Liang S, Somogyi R (2000) Genetic network inference: from co-expression clustering to reverse engineering. Bioinformatics 16:8: 707-26
35. Kauffman SA (1993) The Origins of Order, Self-Organization, and Selection in Evolution. Oxford University Press
36. Kauffman SA (1969) Metabolic stability and epigenesis in randomly connected nets. J Theoret Biol 22:437-467
37. Wang PP, Robinson J (2003) What is SORE? The 7th Proceedings of the Joint Conferences on Information Sciences
38. Wang PP, Cao Y, Robinson J, Tokuta A (2003). A study of two-genes network- The simplest special case of SORE. The 7th Proceedings of the Joint Conferences on Information Science
39. Wang PP, Tao H (2003). A novel method of error correcting code generation based on SORE. The 7th Proceedings of the Joint Conferences on Information Sciences
40. Wang PP, Yu J (2003) SORE, Self Organizable & Regulating Engine- A powerful classifier. The 7th Proceedings of the Joint Conferences on Information Sciences
41. Thieffry D, Huerta AM, Perez-Rueda E, Collado Vides J (1998) From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in Escherichia coli. Bioessays 20:433-440
42. Shmulevich I, Zhang W (2002) Binary analysis and optimization-based normalization of gene expression data. Bioinformatics 18:4:555-565
43. Wolfram S (1983) Cellular Automata. Los Alamos Science 9:2-21
44. Weisstein WE "Limit Cycle." MathWorld—A Wolfram Web Resource. http://mathworld.wolfram.com/Attractor.html
45. Weisstein WE "Limit Cycle." MathWorld—A Wolfram Web Resource. http://mathworld.wolfram.com/BasinofAttraction.html
46. Weisstein WE "Limit Cycle." MathWorld—A Wolfram Web Resource. http://mathworld.wolfram.com/Limitcycle.html
47. Somogyi R, Sniegoski C (1996) Modeling the complexity of genetic networks. Complexity 1:6:45-63

48. Szallasi Z (1999) Genetic network analysis in light of massively parallel biological data acquisition. Pacific Symposium on Biocomputing 4:5-16
49. Nguyen HT, Walker EA (2000) A First Course in Fuzzy Logic. Chapman & Hall/CRC 2-12
50. Reiter CA (2002) Fuzzy Automata and Life. Complexity, 7:3:19-29
51. Wang L (1997) A Course In Fuzzy Systems And Control. Prentice Hall PTR 29-32