Evolving Random Boolean Networks with Genetic Algorithms for Regulatory Networks Reconstruction

Mariana R. Mendoza
PPGC, Instituto de Informática
Universidade Federal do Rio Grande do Sul
C.P. 15064, 91501–970
Porto Alegre, RS, Brazil
mrmendoza@inf.ufrgs.br

Ana L. C. Bazzan
PPGC, Instituto de Informática
Universidade Federal do Rio Grande do Sul
C.P. 15064, 91501–970
Porto Alegre, RS, Brazil
bazzan@inf.ufrgs.br

ABSTRACT

The discovery of the structure of genetic regulatory networks is of great interest for biologists and geneticists due to its pivotal role in organisms' metabolism. In the present paper we aim to investigate the inference power of genetic regulatory networks modeled as random boolean networks without the use of any prior biological information. The solutions space is explored by means of genetic algorithms, whose main goal is to find a consistent network given the target data obtained from biological experiments. We show that this approach succeeds in reconstructing a model with satisfactory level of accuracy, representing an useful tool to guide biologist towards the most probable interactions between the target genes.

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: Biology and genetics

General Terms

Algorithms

Keywords

Gene Regulatory Network, Genetic Algorithm, Boolean Network, Reverse Engineering, Gene Interactions

1. INTRODUCTION

One of the most challenging problems in bioinformatics is the inference of genetic regulatory networks (GRNs). With the recent advances in biological experiments and the availability of large amounts of gene expression data, scientists have shifted their attention to a systematic study of organisms, rather than a gene-by-gene analysis. The main motivation is to gain insight into the underlying complex networks of organisms, as well as to characterize the genetic regulation responsible for cellular development and function.

GECCO'11, July 12–16, 2011, Dublin, Ireland.

The process of identifying gene interactions is commonly referred in literature as reverse engineering of GRNs. There are three important factors for the effectiveness of this procedure, as outlined by Marbach in [10]: the qualitative and quantitative features of gene expression dataset; the selected modeling framework and, at last, search methods applied to find the most probable network structure given the dataset and any available prior knowledge.

Nowadays, most research on the reconstruction of GRNs is based on gene expression data from microarray experiments. The microarray technology allows the measurement of mRNA concentration for a large number of genes. This information can be interpreted as the expression levels of genes under certain environmental conditions and, theoretically, can be used to discover the interactions among genes when a large enough dataset is available. However, this approach suffers from the so-called dimensionality problem: the quantity of time points is usually scarce considering the large number of observed genes [19]. This issue, combined to the fact that gene expression data is intrinsically noisy, is one of the current drawbacks of the reverse engineering process.

In the last years, a wide variety of frameworks to model GRNs have been proposed. In a general way, they can be either continuous or discrete, deterministic or stochastic, static or dynamic, as mentioned in a comparative study of reverse engineering methods held by Hache et al. in [4]. Bayesian networks, for instance, have been applied to GRNs modeling and learning by Friedman et al. in [3]. This is a particularly suitable tool for handling noisy data since the probabilistic nature of Bayesian Networks allows the analysis of the statistical properties of dependence and conditional independence between multiple interacting entities. Also, techniques that cover the continuous level of gene expression, like neural networks and S-systems, have been successfully explored in [12] and [17] respectively. Both methods have the advantage of reproducing the non-linearity of genetic regulation, but require the inference of a large number of parameters.

Regarding the discrete approaches, one of the most well-known methods for modeling GRNs is based on random boolean networks, which were first applied to this context by Kauffman in [5]. Although they are extremely simple, random boolean networks are able to capture much of the complex dynamics of gene networks and allow the extraction of meaningful biological information [8]. Algorithms for inferring gene networks with random boolean networks

have been early proposed in [1] and [9], and more recently in [8] and [11]. These works focus on finding a consistent network given the input data (which has been lately referred as the *consistency problem*), by performing an exhaustive search over the state transition matrix of discretized gene expression data.

As previously mentioned, a good modeling framework does not guarantee the effectiveness of the procedure by itself; it must be combined with efficient search or learning algorithms so that relevant knowledge can be discovered. Lately, evolutionary algorithms (EAs) have attracted researchers attention due to its ability to cope with a large solution space [14]. They have been widely used in genetic data analysis in association with the distinct frameworks formerly presented.

In [2], genetic algorithms are used to optimize the structure of networks modeled by a weight matrix of regulatory pathways. The expressions are regulated in discrete state transitions, given by the weighted sum of expression level of other genes. Solutions were evaluated based in the absolute error between the generated expression and the target expression pattern, and search was biased by the Minimum Description Length so that obtained results are the most simplest structured models. Authors have applied this methodology to randomly generated target networks and have successfully identified networks with several dozen genes to significant accuracy.

Discrete models were also used in [15] to develop a ranking method of alternative hypothesis models to a target GRN. Since experimental data has limited availability, authors propose a more realistic approach, in which the set of alternative hypotheses is given as input file to a genetic algorithm together with the gene expression data. Through the minimization of a quadratic error function between expression profiles obtained from generated and target networks, the method successfully ranked the alternative hypothesis, identifying the most probable network structure given the input gene expression patterns.

Although GRNs may be easily codified into discrete models, most of researches involving EAs still use continuous modeling frameworks. S-Systems, one of the most common approaches, have been combined with EAs in [13] and [18], providing satisfactory reconstruction for small and middle-sized genetic regulatory networks. In [7], a hybrid system has been proposed, combining genetic algorithms with a single-layer artificial neural network. The algorithm has accurately fit the data on which is was trained for both artificial and real data, and correctly reproduced connections between genes when trained with artificial data. For further information on different types of EAs for reverse engineering GRNs using S-Systems and neural networks as modeling frameworks, see [17]. Also, a broad review of applications of EAs in different problems of bioinformatics can be found in [14].

In the present work we aim to investigate the viability of reverse engineering GRNs, modeled as random boolean networks, by exploring the search space of consistent topologies with genetic algorithms based solely in experimental data. Our goal is to evaluate the power of inference of this approach and how far we can reconstruct an accurate model without supplying any biological prior knowledge. This is useful when no such prior knowledge is available, which is a common situation. The generated model would be a valuable start point for biologists in the investigation of gene interactions.

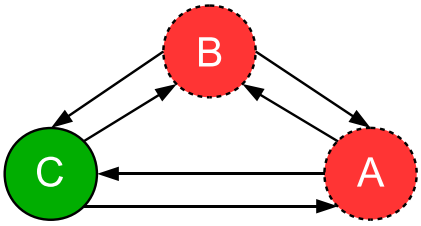

Figure 1: An example of $N = 3$ interacting genes, with $K = 2$, modeled as boolean devices. A green node represents expressed genes (state 1), while a red dashed node denotes not expressed ones (state 0).

This paper is organized as follows. In the next section we will briefly describe the RBN-based modeling framework and its dynamics. In the sequence, we present a detailed description of the proposed model. In section 4 the data and parameters values used in experiments are explained. Finally, we present and discuss the results of our work, as well as possibilities of improvements and further research.

2. METHODS: RANDOM BOOLEAN NETWORKS

A random boolean network (RBN), $G(V, F)$, is defined by a set of nodes $V = \{x_1, x_2, \ldots, x_N\}$, which in GRNs context represent genes, and a set of boolean functions $F = \{f_1, f_2, \ldots, f_N\}$. Each node x_i, $i = 1, \ldots, N$, is a boolean device that stands for the value (state) of gene i. Generally, $x_i = 1$ denotes that gene i is expressed, while $x_i = 0$ means that it is not expressed. Each node has its value determined by a boolean function $f_i \in F$, which represents the rules of regulatory interactions between genes, and K_i specific inputs (x_j, with $j = 1, \ldots, N$), denoting its regulatory factors.

Function f_i specifies, for each possible combination of K_i input values, the status of the regulated variable x_i. Thus, being K_i the number of input variables regulating a given gene, since each of these inputs can be either expressed or not (1 or 0), the number of combinations of states of the K_i inputs is 2^{K_i}. For each of these combinations, a specific boolean function must output either 1 or 0. Therefore the total number of boolean functions over K_i inputs is $2^{2^{K_i}}$. When $K_i = 2$, some of these functions are well-known (AND, OR, XOR, NAND, etc.), but in the general case functions have no obvious semantics.

To illustrate the regulation process, Figure 1 depicts a simple example of a GRN modeled as a RBN of $N = 3$ genes controlled by 2 regulatory factors each ($K_i = 2$). The

Table 1: Boolean functions for genes A, B, and C.

(OR)			(OR)			(NAND)		
B	C	A	A	C	B	A	B	C
0	0	0	0	0	0	0	0	1
0	1	1	0	1	1	0	1	1
1	0	1	1	0	1	1	0	1
1	1	1	1	1	1	1	1	0

Table 2: State transition for Table 1.

(t)			(t+1)		
A	B	C	A	B	C
0	0	0	0	0	1
0	0	1	1	1	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	1	1	0
1	1	1	1	1	0

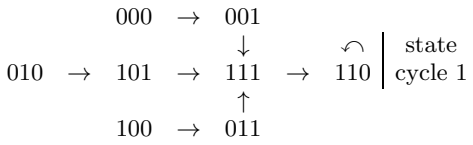

Figure 2: State transition graph for Table 1.

parents's set for nodes A, B and C are, respectively, {B,C}, {A,C} and {A,B}. A boolean function is randomly assigned to each gene and the final regulatory rules are shown in Table 1: genes A and B are regulated by function OR, while gene C is regulated by a NAND function. Given the boolean functions from Table 1, Table 2 shows, for all 2^3 states at a given time t, the resulting expression of each gene x_i at time $t+1$, i.e. the successor state of each state. Further, from this table, it is possible to determine the state transition graph of the network, which appears in Figure 2. One sees that there is only one attractor state for this example, namely `110`.

Random boolean networks have been used to explain adaptation and self-organization in complex systems. The study of the behavior of regulatory systems by means of networks of boolean functions was introduced by Kauffman in 1969 [5]. Examples of the use of this approach in biology, genomics, and other complex systems can be found in [6].

3. PROPOSED MODEL

3.1 Coding description

Due to the interesting combination of simplicity and inference capacity of RBNs, this was the selected framework to describe the structure and dynamics of regulatory networks. As previously mentioned, in this context each node of the network represents a gene and the edges translate the existing dependencies between them. Each gene might be either at state 1 or 0, indicating if it is expressed or not, respectively. A gene is considered expressed when the protein which it codes is being produced. The regulation of gene expression is controlled by other genes, henceforth referred as gene's parents, through boolean functions. The boolean function denotes the necessary conditions in terms of parents' state for a gene to express its carried information.

As we are interested in reconstructing the conditions and relations of gene regulation from experimental data, this is an adequate model because it allows us to concentrate in the qualitative features of the network. Former works have tackled the task of discovering GRNs by performing an exhaustive search over states transition table of experimental data or discretized gene expression data [1, 9, 11]. This kind of search has succeed in finding most of topology of the underlying target GRN, but it requires high computational time even for small-sized networks. Since most organisms are composed of hundreds of genes at least, unveiling the corresponding GRNs would require a prohibitive amount of time, if feasible at all. Therefore, more powerful search methods have been recently proposed. The use of genetic algorithms (GAs) is useful because they are known to cope well with a large solution space and they have a large amount of implicit parallelism.Thus, GAs have been widely applied in different tasks of genetic analysis, as reviewed in [14]. In the present work, we combine the robustness of GA with the simplicity of RBNs aiming to reach an accurate reconstruction of GRNs.

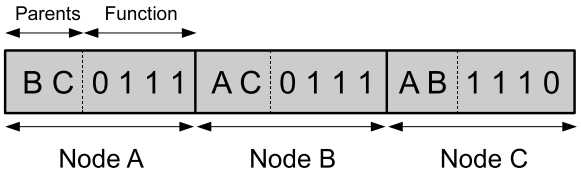

Figure 3: An example of GA individual codification for network depicted in Figure 1.

In the GA-based modeling, each individual represents a candidate network, whose topology is codified into a binary string. This string contains the binary values of parents' IDs and the boolean function for all genes in the network. Genes are identified from 1 to N, where N is the number of genes in GRNs, and have, each, K_{max} or less connections. Both N and K_{max} are user-configurable parameters.

An example of codification is given in Figure 3 for the RBN depicted in Figure 1 and its corresponding boolean functions in Table 1. For illustration purposes, this example shows genes identified as A, B and C. However, in the real application, these IDs are binary coded. Also, note that only the function's output values of Table 1 are represented in the string (the last four bits of each node's encoding is reserved for its *function* in Figure 3). The information in the bits referring to the gene's parents is the key to retrieve the input combination. This mechanism avoids representing all possible input/output combinations, which might get excessively large for higher values of K. The length of binary strings clearly depends on the number of genes and the maximum number of allowed connections per gene (K_{max}). For large-sized networks, the strings may end up being long. However, this has not posed any problem related to computational time.

3.2 Fitness function

Individuals are evaluated based in a so-called inconsistency ratio (IR) computed from a target data. Each measurement in the given data set is associated with a positive weight which indicates its quality. Since it is usually impracticable to estimate the quality of each measurement, we assume weights to be uniform and equal to 1, following the suggestion in [11]. For each 2^K possible input combination of a node, $k = 1, ..., 2^K$, the total weight of measurements whose output value is 0 and 1 are stored in variables $w_k(0)$ and $w_k(1)$, respectively. Considering that w is the sum of

Table 3: Example of inconsistent data.

Input	x_{i_1}	1	0	1	0	0	1	0	0	1	1
	x_{i_2}	0	0	1	1	0	0	1	0	0	1
Output	x_i	0	1	0	1	1	1	1	0	0	0

all measurements' weights, the IR for a single node is hence defined as in Equation 1.

$$IR_i = w^{-1} \sum_{k=1}^{2^K} min(w_k(0), w_k(1)) \quad (1)$$

Inconsistencies are related to the number of mismatches found in the gene expression profiles in respect to the structure of networks generated by the GA. For a given combination of input, we expect the model to generate the same output according to the regulation rules described by the boolean functions. Otherwise, an inconsistency exists. For example, in Table 3, inputs $(0,0)$ and $(1,0)$ produce respectively outputs 1 and 0 in most of experiments. Therefore, whenever the output for these inputs are 0 and 1, these are considered inconsistent values. The inconsistency ratio for this specific node and the given data is equal to $2/10 = 0.2$. In [11] this criteria was used to find the most consistent parent set and boolean function for each gene, since the noise in experimental data renders the identification of a single consistent tuple and mapping rule not feasible.

Once the IR for each gene is calculated, the network inconsistency is determined by the sum of all nodes' IR, according to the following equation:

$$IRN = \sum_{i=1}^{N} IR_i \quad (2)$$

The goal of the genetic algorithm is to minimize the network inconsistency regarding the input expression profiles. The fitness function is thus defined such that the least inconsistent individuals are more likely to be selected in further generations:

$$\phi_1 = \frac{1}{1 + \frac{IRN}{N \times 0.5}} \quad (3)$$

in which $N \times 0.5$ refers to the maximum inconsistency value that may be carried by a network.

In order to bias our search towards sparser networks, which are known to be GRNs' representative, we also tested the inclusion of a penalty factor in the fitness function. This factor is computed as the number of inferred pathways in the model (NP) divided by the maximum number of possible connections, as shown in Equation 4.

$$\phi_2 = \frac{1}{1 + \frac{IRN}{N \times 0.5} + \frac{NP}{N^2}} \quad (4)$$

3.3 Algorithm

An overview of the proposed method is depicted in Figure 4. Also, a general view of the implemented GA is shown in Algorithm 1. The first stage consists of setting the parameters for the algorithm and randomly generating the initial population (networks). Once created, individuals must

Table 4: Inconsistency analysis for data in Table 3.

k	Input	$w_k(0)$	$w_k(1)$	$min(w_k(0), w_k(1))$
1	(0,0)	1	2	1
2	(0,1)	0	2	0
3	(1,0)	2	1	1
4	(1,1)	2	0	0

$$IR_i = \frac{1+0+1+0}{10} = 0.2$$

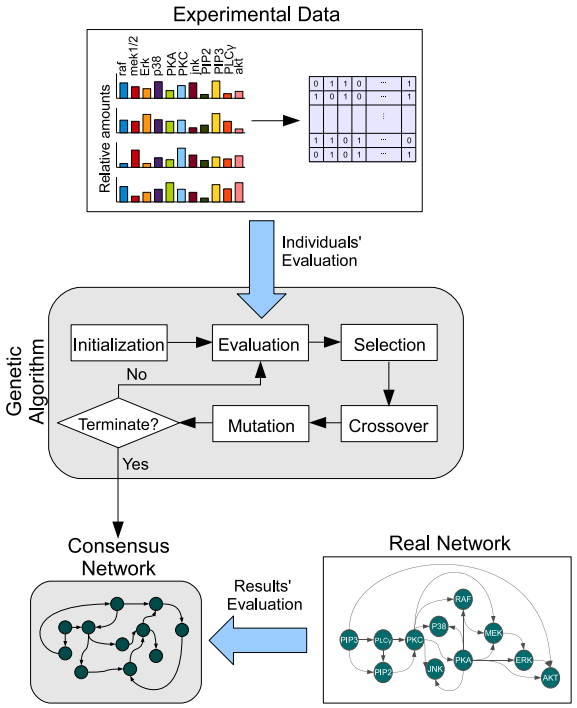

Figure 4: An overview of the implemented method.

be evaluated according to a fitness function so that selection can happen. In the present work the evaluation criteria is related to the minimization of an inconsistency ratio regarding a given gene expression data set, which will be later explained in Section 4.1. Each individual is decoded and the GRN topology is saved as an adjacency matrix. Then, the inconsistency ratio is computed for each network's node in respect to its parents' set.

After the fitness value has been computed for all individuals, elitism is applied. In this phase, the E fittest individuals are selected to integrate the next generation without suffering any change in their genetic material. The individuals to fill the remaining slots in the subsequent generation are selected, in proportion to their fitness values.

The group of individuals on which crossover and mutation operators are applied is the pool generated by the selection mechanism. Individuals are recombined in pairs with probability P_{cross} using two-point crossover. Each bit suffers mutation with probability P_{mut}. In order to allow further variability in the first generations, this probability is initialized with a high value and is gradually decreased by a constant factor until it reaches an user-configurable target.

As GAs provide us a set of candidate solutions ranked by their fitness, at the end of the simulation, it is necessary to

somehow choose a network that best represents the hidden relations of the input biological data based in the supplied solutions set. One possible approach is to apply the concept of *wisdom of crowds*, where the information contained within each individual is combined in order to make more accurate network predictions. A simple, but still widely used technique to do so is known as majority voting: every network in the best solutions set votes on the classification of each edge (interaction) as present or absent. The existence of the interaction is then defined according to the majority of the votes.

In this work we adopt a similar approach, creating a consensus network based on individuals of the last generation. Although they may have different fitness values, all of them carry important information since they explore different sites of the solutions space. The first step towards the consensus network consists of generating a confidence matrix, which reflects the portion of individuals in the solutions set in which a certain interaction exists. A consensus network is then created by including all the edges whose corresponding factor (confidence level) in the confidence matrix is superior than an user-configurable threshold. This process is repeated for each simulation run. In the end, the final consensus network is created aby pplying the same methodology to all previously found partial consensus networks, as shown in Figure 5.

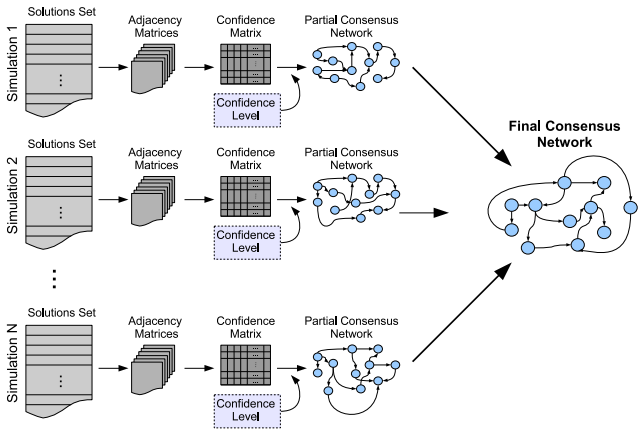

Figure 5: Identification process of the consensus network.

4. EXPERIMENTS

4.1 Data

The gene expression profiles used in this work derive from intracellular multicolor flow cytometry experiments applied by Sachs in [16] over eleven phosphorylated proteins and phospholipids that compose the Raf pathway. Raf is a critical signalling pathway involved in the control of cell proliferation in human immune system cells. A deregulation of this pathway may lead to carcinogenesis. Therefore, this network has been extensively studied and a representation of the currently accepted gold standard network is available, which can be seen in Figure 6.

At this point we stress that we use this standard network just for evaluation purposes. We have not used the structure or any other knowledge about the real network in the proposed method. Rather, the main motivation is exactly to avoid such use, employing only the experimental data in the inference problem, as will be later explained in Section 5.

Algorithm 1 General view of the implemented GA.

```
Initialize parameters and first generation
for each generation do
  for each individual do
    Decode network's structure
    Compute fitness of each individual
  end for
  Create a new population of P_size where the E fittest remain and the others are selected and reproduced based on their fitness
  Apply point mutation with probability P_mut to individuals not in elite
end for
Find consensus network
```

The experiments run by [16] resulted in 5400 data points, where 1200 are observational and 4200 are interventional. In [20] the original data was randomly sampled to smaller data sets so that they would represent a better figure of microarray experiments, which do not provide such abundance of data. In this process, five data sets of 100 measurements each were originated from the observational data.

Discretization of the reduced observational data sets into binary values was performed based in the median, as suggested in [21]. The two smallest and largest values were considered outliers and thus discarded. Assuming that measurements are disposed in a $r \times c$ matrix (A), where rows contain experiments results and each column refers to a different gene, the median for each protein is computed (M_C). The upper 50 percentile was treated as expressed genes (1) and the lower 50 percentile as unexpressed genes (0). Is important to mention that the term *gene* is generically used to denote all interacting nodes in the network, albeit they may actually refer to genes' products, such as proteins.

4.2 Parameters

The GA was implemented with a population of $P_{size} = 50$ individuals evolved for 1000 generations. Elitism is applied with an elite size $E = 4$. As previously mentioned in Section 3.3, the mutation probability starts with a high value and is gradually decreased. The initial probability is configured as 0.1 and it is decreased until it reaches the

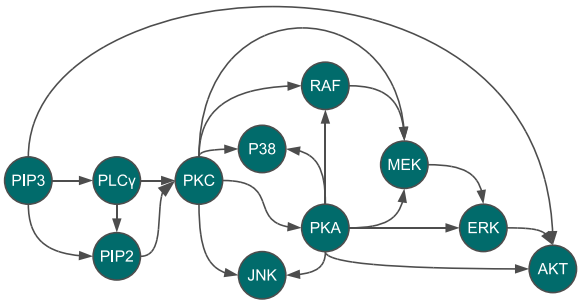

Figure 6: Raf signalling pathway.

Table 5: GA parameters.

Parameter	Description	Value
G_{max}	Number of generations	1000
P_{size}	Population size	50
E	Elite size	4
$P_{mut_{init}}$	Initial mutation probability	0.1
$P_{mut_{min}}$	Minimum mutation probability	0.001
P_{cross}	Crossover probability	1

limit value of 0.001. Crossover is applied with probability 1. These values were chosen based on previous experiments.

According to the target network, depicted in Figure 6, we set up N to 11 genes and tested the model for K_{max} equal to 2 and 3. K_{max} represents the maximum connectivity of a single node, which means that the generated model has the flexibility of having a node with less connections if this simplification results in a better description of input gene expression patterns. However, self-loops are not allowed since they will always be perfectly consistent according to consistency problem and therefore, search would be biased towards topologies with high frequency of self-loops. All parameters' values are summarized in Table 5.

5. RESULTS

To test the effectiveness of the proposed method, 30 simulations were run and the results were then combined into a consensus network, as explained in Section 3.3. The simulations were set up by varying K_{max} value between 2 and 3. Also, the penalty factor in the fitness function was either included or not. The algorithm was implemented in MATLAB and it requires approximately 5 minutes to perform all the evolution for a single simulation in a PC with a Core 2 Duo T7500 processor and 2Gb RAM memory.

Our method has two evaluation points, as shown in Figure 4. The first one refers to the scoring process of GA individuals applying the fitness functions previously defined (Equations 3 and 4). This step receives as input a random subset of the observational data sets generated by [20], discretized as explained in Section 4.1. The subset is a random choice in two senses: at each evaluation, one of the five data sets is selected with equal probability and then a sequence of 30 measurements is randomly chosen. As these data points have been already sampled from a larger collection, they do not hold any time dependency between them. This procedure aims in preventing the model of overfitting the data.

The second evaluation point occurs at the end of the algorithm's execution and consists of assessing the method's performance. A confusion matrix is created by comparing the structure of both GA generated and real networks. This matrix quantifies the inferred correct interactions (true positives, TP), incorrect interactions (false positives, FP), correct non-interactions (true negatives, TN) and incorrect non-interactions (false negatives, FN). Accuracy and precision are then calculated according to Equations 5 and 6, respectively:

$$Accuracy = \frac{TP + TN}{n} \tag{5}$$

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

Table 6: Average accuracy and precision over 30 runs for a confidence level of 95%.

K_{max}	Penalty Factor?	Accuracy		Precision	
		avg.	std.	avg.	std.
2	No	0.806	0.018	0.335	0.110
	Yes	0.816	0.020	0.386	0.126
3	No	0.781	0.022	0.286	0.083
	Yes	0.797	0.024	0.325	0.113

Table 7: Average number of true positives edges in partial consensus networks for simulations including the penalty factor.

K_{max}	True Positive Edges	
	avg.	std.
2	3.5	1.1
3	4	1.5

In Equation 5, n denotes the number of maximum possible edges, i.e. $n = TP + TN + FP + FN$. The accuracy metric reports the degree to which information on the inferred model matches the target network. Precision reflects the relevant proportion of the total predicted edges and is useful to find how similar are inferred networks to each other.

The obtained results are summarized in Table 6. These metrics are drawn from the partial consensus networks generated for each execution of the GA. We applied a confidence level of 95%, which means that an edge will be included in the partial consensus network only if it occurs in at least 95% of the solutions indicated by the final generation. As one can see at Table 6, results concerning accuracy were quite satisfactory. The applied approach was able to reconstruct the Raf signalling pathway with an average accuracy of 0.8. This is a satisfactory score considering no prior knowledge about the network structure has been supplied to the algorithm. The inference has been made solely based upon discretized gene expression data.

The highest accuracy values are related to simulations with $K_{max} = 2$: the measured accuracy was 0.806 for simulations run with fitness function of Equation 3, and 0.816 for simulations with the inclusion of the penalty factor. For this configuration, partial consensus networks have on average 3.5 correctly inferred interactions, with a standard deviation of 1.1. The occurrence of true positive edges increases for $K_{max} = 3$, in which 4 true interactions are inferred per partial consensus network on average, with a standard deviation of 1.5. However, as the frequency of false positives is also higher for the case where $K_{max} = 3$, there is no performance improvement. In fact, accuracy is lower in simulations with $K_{max} = 3$. Also, the inclusion of a penalty factor to prioritize the evolution of sparser topologies seems to improve both accuracy and precision means.

Unlike accuracy, precision values are low. The average precision for simulations made is 0.33. Here, we formulate two possible reasons. First, the fact that partial consensus networks contain several amount of false positive influences the precision metric (see Equation 6). Also, we understand that this may be produced by the stochastic nature of GA, which allows individual solutions to be different from each other aiming to better explore the search space. This can be verified by the fact that although partial consensus networks

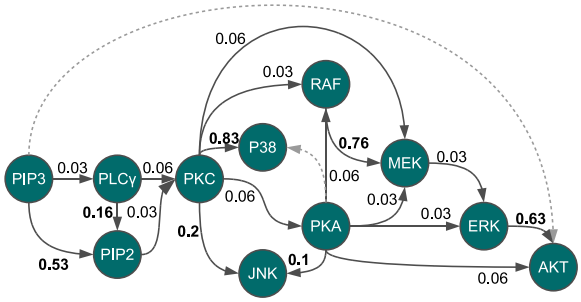

Figure 7: Confidence level associated to edges in the final consensus network for simulations with $K_{max} = 2$. Edges drawn in gray dotted line were not inferred by the method.

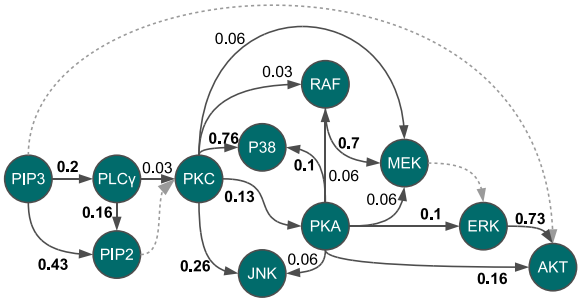

Figure 8: Confidence level associated to edges in the final consensus network for simulations with $K_{max} = 3$. Edges drawn in gray dotted line were not inferred by the method.

have few true positive edges, when the 30 partial solutions are combined into one final consensus network, most of the existing interactions are correctly inferred.

Figures 7 and 8 show the confidence levels associated to edges of the RAF signalling pathway in the final consensus network of simulations with $K_{max} = 2$ and $K_{max} = 3$, respectively, and including the penalty factor. These values are obtained after the combination of the partial consensus networks and reflect the probability of occurrence of a given interaction. The gray dotted lines denote interactions which were not inferred by the proposed model. Also, edges whose confidence level are greater than 0.1 are highlighted with the corresponding numbers written in boldface.

In Figure 7 it is possible to observe that the method correctly predicts 18 out of the 20 true interactions in the RAF signalling pathway when $K_{max} = 2$. From these, 7 have appeared in at least 10% of the partial consensus networks and are therefore associated to higher confidence levels. Both PIP3 $\rightarrow$ AKT and PKA $\rightarrow$ P38 interactions have not been inferred. The shown values refer to the method's raw results. Since we aim to analyse how powerful is the implemented algorithm in predicting the existing interactions, no filtering mechanism have been applied in the final network construction , i.e., edges have not been discarded according to a given threshold so that only the most probable ones remain in the model.

The execution of the algorithm with $K_{max} = 3$ has correctly inferred 17 existing connections. Although the accuracy is lower than simulations with $K_{max} = 2$, these simulations resulted in a larger set of interactions associated with higher confidence levels. In this case, 11 out of the 17 inferred connections exists in 10% of solutions or more. The missing edges are PIP3 $\rightarrow$ AKT, PIP2 $\rightarrow$ PKC and MEK $\rightarrow$ ERK.

6. CONCLUSION

Nowadays, reverse engineering of genetic regulatory networks is a major challenge in bioinformatics. The relevance of these networks for organisms' metabolism brings a wide range of knowledge application, such as the development of new drugs and treatments. Despite the great interest of scientists and the availability of high quality data, achievements are still modest and proposed methods have not been able to fully infer GRNs to high accuracy levels yet. Most of available reverse engineering methods reach good results by either applying some source of biological knowledge in the inference process or performing an exhaustive search over the solutions space. However, neither of these scenarios are realistic or desirable, since both biological knowledge and time are usually limited.

In the present paper we combined RBNs and genetic algorithms for the reconstruction of GRNs without the inclusion of any biological prior knowledge. The inference is based solely in experimental data and information about real GRN's structure was used only for performance assessment. The target network is the Raf signalling pathway, which is involved in the regulation of cell proliferation in human immune system cells. Simulations were run varying the number of maximum connections per node, given by parameter K_{max}, and by either including or not a penalty factor into the fitness function. This factor intends to prioritize sparser topologies, which are known to be GRNs' representatives. Each condition was simulated 30 times and concepts of *wisdom of crowds* were applied to find partial consensus networks based in all individuals of the last generation of the GA, as explained in Section 3.3. A final consensus network is created at the end of the process to summarize the results of the simulations set.

Regarding the reconstruction of the Raf pathway, the implemented method reached a good accuracy level considering that no biological information was provided. All regulatory factors and rules were inferred from discretized experimental data solely. The average accuracy for the simulations run is 0.8. However, the average precision reached a much lower mark: 0.33. We explain this in terms of the high number of false positives and the stochastic nature of GA, which allows individuals to explore different sites of search space and, therefore, to have different topologies between themselves. However, exploring different solutions simultaneously is extremely advantageous when the partial consensus networks are combined into a final consensus network: almost all interactions in the Raf signalling pathway have been correctly inferred by the model. For simulations with $K_{max} = 2$, 18 out of the 20 existing interactions in the Raf signalling pathway were correctly predicted, while for $K_{max} = 3$, this number reduced to 17.

We found the accuracy of the generated model to be satisfactory considering the several constraints involved in the problem of inferring GRNs. Although occurrence probability may be low for some interactions, their presence in the

final consensus network is a relevant evidence to suggest further investigation towards these interactions through in vitro experiments. However, generated models contain many false positives that must be filtered out in order to improve both accuracy and precision.

Therefore, for future works, we deem advisable to study filtering techniques robust enough to successfully differentiate false positives from true positives associated to low confidence levels. It would be equally interesting to test the effect of distinct thresholds on the method's performance (accuracy and precision). Another important research direction regards the application of the method to artificial networks of different dimensions so that reconstruction efficiency can be evaluated in scenarios of distinct complexity. Finally, we stress the necessity of studying a fitness function efficient in coping with self-loops, which are known to be prominent in real GRNs. We believe that these refinements would increase the reliability of information contained in the generated model and thus make it a more relevant source of knowledge for the discovery of GRNs.

7. ACKNOWLEDGMENTS

We would like to thank Dr. Adriano Werhli for the valuable discussions during the development of this work and the anonymous reviewers for suggesting interesting research directions. Both authors are supported by CNPq.

8. REFERENCES

[1] T. Akutsu, S. Miyano, and K. S. Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 17–28, 1999.

[2] S. Ando and H. Iba. Inference of gene regulatory model by genetic algorithms. In *Proceedings of the 2001 IEEE Congress on Evolutionary Computation*, pages 712–719. IEEE CS, 2001.

[3] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using bayesian networks to analyse expression data. In *Journal of Computational Biology*, volume 7, pages 601–620. Mary Ann Liebert, Inc., 2000.

[4] H. Hache, H. Lehrach, and R. Herwig. Reverse engineering of gene regulatory networks: A comparative study. In *EURASIP Journal on Bioinformatics and Systems Biology*, volume 2009, pages 8:1–8:12. Hindawi Publishing Corp., 2009.

[5] S. A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *J Theor Biol*, 22(3):437–467, March 1969.

[6] S. A. Kauffman. *The Origins of Order*. Oxford University Press, Oxford, 1993.

[7] E. Keedwell and A. Narayanan. Discovering gene networks with a neural-genetic hybrid. In *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, volume 2, pages 231–242. IEEE CS, 2005.

[8] H. Lähdesmäki, I. Shmulevich, and O. Yli-Harja. On learning gene regulatory networks under the boolean network model. In *Machine Learning*, volume 52, pages 147–167. Kluwer Academic Publishers, 2003.

[9] S. Liang, S. Fuhrman, and R. Somogyi. REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 1998, pages 18–29, 1998.

[10] D. Marbach. *Evolutionary Reverse Engineering of Gene Networks*. PhD thesis, École Polytechnique Fédérale de Laussane, 2009.

[11] D. Nam, S. Seo, and S. Kim. An efficient top-down search algorithm for learning boolean networks of gene expression. In *Machine Learning*, volume 65, pages 229–245. Springer Science, 2006.

[12] A. Narayanan, E. C. Keedwell, J. Gamalielsson, and S. Tatineni. Single-layer artificial neural networks for gene expression analysis. In *Neurocomputing*, volume 61, pages 217–240. Elsevier, 2004.

[13] N. Noman and H. Iba. Inference of gene regulatory networks using s-system and differential evolution. In *Proceedings of the 2005 Genetic and Evolutionary Computation Congress*, pages 439–446. ACM, 2005.

[14] S. K. Pal, S. Bandyopadhyay, and S. S. Ray. Evolutionary computation in bioinformatics: A review. In *IEEE Transactions on Systems, Man, And Cybernetics*, volume 36, pages 601–615. IEEE CS, 2006.

[15] D. Repsilber, H. Liljenstrom, and A. S. G. E. Reverse engineering of regulatory networks: simulation studies on a genetic algorithm approach for ranking hypotheses. In *BioSystems*, volume 66, pages 31–41. Elsevier, 2002.

[16] K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan. Causal protein-signaling networks derived from multiparameter single-celldata. In *Science*, volume 308, pages 523–529. April 2005.

[17] A. Sîrbu, H. J. Ruskin, and M. Crane. Comparison of evolutionary algorithms in gene regulatory network model inference. In *BMC Bioinformatics*, volume 11, page 59. BioMed Central, 2010.

[18] C. Spieth, F. Streichert, N. Speer, and A. Zell. Optimizing topology and parameters of gene regulatory network models from time-series experiments. In *Proceedings of the 2004 Genetic and Evolutionary Computation Congress*, pages 461–470. Springer-Verlag, 2004.

[19] Y. Wang, T. Joshi, X. Zhang, D. Xu, and L. Chen. Inferring gene regulatory networks from multiple microarray datasets. In *Bioinformatics*, volume 22, pages 2413–2420. Oxford University Press, 2006.

[20] A. V. Werhli and D. Husmeier. Gene regulatory network reconstruction by bayesian integration of prior knowledge and/or different experimental conditions. In *Journal of Bioinformatics and Computational Biology*, volume 6, pages 543–572. 2008.

[21] W. Zhao, S. Erchin, and E. R. Dougherty. Inferring gene regulatory networks from time series data using the minimum description length principle. In *Bioinformatics*, volume 22, pages 2129–2135. 2006.