An Epsilon-Greedy Mutation Operator Based on Prior Knowledge for GA Convergence and Accuracy Improvement: an Application to Networks Inference

Mariana R. Mendoza
PPGC, Instituto de Informática
UFRGS, Porto Alegre, RS, Brazil
mrmendoza@inf.ufrgs.br

Adriano V. Werhli
Centro de Ciências Computacionais
FURG, Rio Grande, RS, Brazil
werhli@gmail.com

Ana L. C. Bazzan
PPGC, Instituto de Informática
UFRGS, Porto Alegre, RS, Brazil
bazzan@inf.ufrgs.br

***Abstract*—This paper introduces a new mutation operator for networks inference based on the epsilon-greedy strategy. Given some prior knowledge, either provided by a third party method or collected from literature, our approach performs mutations by randomly exploring the search space with ϵ-frequency and by exploiting the available prior knowledge in the remaining cases. The algorithm starts with a highly exploitative profile and gradually decreases the probability of employing prior knowledge in the mutation operator, thus reaching a trade-off between exploration and exploitation. Tests performed have shown that the proposed approach has great potential when compared to the traditional genetic algorithm: it not only outperforms the latter in terms of results accuracy, but also accelerates its convergence and allows user to control the evolvability speed by adjusting the rate with which the probability of using prior knowledge is decreased.**

***Keywords*-Genetic Algorithm; Mutation; Epsilon-Greedy; Prior Knowledge; Mutual Information; Networks Inference**

I. Introduction

Genetic algorithms (GA) [1] are a class of evolutionary algorithms that aims at optimizing complex computational problems using techniques based on the natural evolution theory. In contrast to deterministic optimization methods, which tend to get trapped in local optima, GA perform a global stochastic search that parallel evolves a population of potential solutions by selectively composing subsequent generations through bio-inspired mechanisms of crossover and mutation. Due to their outstanding performance on real, hard problems, as well as to their great versatility regarding solutions representation, GA have received great attention from the scientific community in optimization tasks [2]. Recent applications include, but are not limited to, parameters learning in neural networks [3], RNA secondary structure prediction [4] and networks inference ([5], [6]).

In the last decades, a wide range of enhancements have been proposed and investigated, most of them concentrated on more effective crossover operators [7]. However, mutation also plays a substantial role in improving GA performance, thus motivating recent efforts towards the design of new mutation operators. Within this context, recent works ([7], [8], [9], [10]) have proposed distinct strategies to either compute the mutation probability or to accept a mutation proposal, some of which are based on optimization mechanisms such as Simulated Annealing (SA) and Softmax. In [8], for instance, SA was combined with GA in order to dynamically change the probability of accepting some inferior solutions: after mutation occurs, the generated solution is evaluated and SA is applied to decide whether to accept it or to keep the previous solution.

In contrast, in [10] authors proposed to apply Softmax to compute the mutation probability of each bit, replacing the traditional blind, random mutation. In this approach, the top and bottom n solutions are taken as positive and negative examples, respectively, and the Boltzmann probability distribution is used to determine the probability of each bit to assume value 0 or 1 in the next generation according to these extreme examples. It has been shown that the Softmax mutation operator causes a faster evolution than the traditional approach, allowing the control of the evolutionary speed by means of the parameter used as base in the probability formula. Furthermore, in [9], the mutation probability was dynamically adapted according to individuals' fitness: an exponential decrease was applied such that high-fitness solutions are protected, while solutions with subaverage fitness are disrupted.

However, to our knowledge, none of the improvements proposed so far have employed prior knowledge to compute mutation probabilities nor applied an epsilon-greedy strategy to control the rate with which the use of prior knowledge is alternated with random operations. The epsilon-greedy approach is broadly used in learning and optimization problems, such as the multi-armed bandit problem [11], to achieve a trade-off between exploration and exploitation and thus improve results accuracy. In short, this method chooses a random option with ϵ-frequency, and otherwise chooses the best available option. Although extremely simple, the epsilon-greedy strategy tends to be hard to beat and significantly better than other optimization methods [11].

In the current work, we propose to balance among mutation decisions made by a traditional blind, random mutation operator and based on some available prior knowledge by means of an epsilon-greedy strategy. Our approach randomly explores the search space in ϵ mutations, and exploits the in-

1522-4899/12 $26.00 © 2012 IEEE
DOI 10.1109/SBRN.2012.40

formation provided by a third party method in the remaining cases. Here, we follow the tendency of combining multiple learners and propose the integration of two distinct inference methods, namely genetic algorithms and mutual information, neither in a parallel or a sequential way, but rather alternating and balancing their use through an epsilon-greedy approach. In the scope of this study, we use mutual information as the source of prior knowledge, albeit any other method, as well as prior knowledge gathered from literature, may be used instead. We validate our mutation operator with the problem of networks inference, more specifically with structure learning in Boolean networks, and show that the coupling scheme proposed in the current work has great potential for inference improvement, outperforming both of the individual methods.

In what follows we introduce our methods and briefly explain the GA modeling and implementation adopted to test our approach. Next, we present the new mutation operator proposed in the current paper and the experiments run to assess it. Finally, we conclude this paper with a discussion of the results and findings of our work.

II. Materials and Methods

A. Random Boolean Networks

Random Boolean networks (RBNs) are defined by a directed graph $G(V, F)$, which is composed by a set of nodes $V = \{v_1, \ldots, v_N\}$, and a set of Boolean functions $F = \{f_1, \ldots, f_N\}$. Each node v_i, $i = 1, \ldots, N$, is a Boolean device that stands for the state of variable i: it can assume values 0/1, on/off, etc. An example of a simple Boolean network structure is depicted in Fig. 1, where the double line denotes a node on state 1, while the single dashed line stands for nodes on state 0.

Each node has its value determined by a Boolean function $f_i \in F$, which represents the rules of regulatory interactions between nodes, and K_i specific inputs, denoting its regulatory factors or *predictors*. Fig. 2(a) specifies the Boolean functions for each node of the example network in Fig. 1. A function f_i determines, for each possible combination of K_i input values, the state of the variable v_i. Being K_i the number of input binary variables regulating a given node, the number of combinations of states of the K_i inputs is

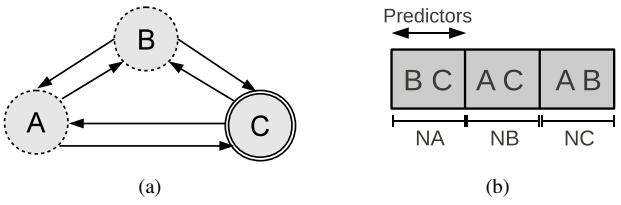

Figure 1. (a) An example of $N = 3$ interacting nodes with $K = 2$ modeled as Boolean devices. Double line nodes assume value 1, while single dashed line nodes are on state 0. (b) The network wiring can be also described as a string with length $N \times K$, containing all nodes' predictors.

A (OR)			B (OR)			C (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

(a)

(t)			(t+1)			(t+2)		
A	B	C	A	B	C	A	B	C
0	0	0	0	0	1	1	1	1
0	0	1	1	1	1	1	1	0
0	1	0	1	0	1	1	1	1
0	1	1	1	1	1	1	1	0
1	0	0	0	1	1	1	1	1
1	0	1	1	1	1	1	1	0
1	1	0	1	1	0	1	1	0
1	1	1	1	1	0	1	1	0

(b)

Figure 2. (a) Boolean functions and (b) a three-step state transition table for the network example of Fig. 1(a).

2^{K_i}. Furthermore, for each of these combinations, a specific Boolean function must output either 1 or 0 such that 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. Given the values of nodes V at time t, the Boolean functions are used to synchronously update the values of nodes at time $t + 1$ by mapping $v_i(t + 1) = f_i(v_{Ki}(t))$, as shown in Fig. 2(b).

B. Mutual Information as Prior Knowledge

In the context of inferring networks structure, information-theoretic approaches such as entropy and mutual information (MI) have been frequently explored as criterion functions ([12], [13], [14]), i.e., a function that evaluates the suitability of a subset of predictors for predicting a target node. In short, this strategy aims at generalizing the pairwise correlation coefficient to measure the degree of independence between two variables. Thus, for each pair of nodes i and j, the mutual information MI_{ij} is computed as [14]:

$$M_{ij} = H_i + H_j - H_{ij} \tag{1}$$

where H_x is the entropy of an arbitrary discrete variable x:

$$H_x = -\sum_{k=1} p(x_k) log(p(x_k)) \tag{2}$$

In (2), $p(x_k) = Prob(x = x_k)$ is the probability of each discrete state (value) of the variable x. It is assumed that a non-zero MI indicates the existence of a relationship among nodes [13]. However, it is important to note that this criterion does not imply a direct causal interaction between these nodes in the real network, but rather that they have a statistical dependence among them, i.e., they are not randomly associated to each other. A common approach for networks inference based on information-theoretic methods is to compute the pairwise MI and apply a threshold such that only those nodes that were linked to others with a MI

higher than the threshold are included in the model ([13], [14]). Since MI is symmetric, this strategy generates an undirected graph G.

Yet, differently from previous works ([13], [14]), we adopt MI as a prior information concerning the target network and use it to support the inference process and improve convergence. The normalized MI matrix obtained from the data is thus interpreted as a degree of belief regarding a relationship among nodes i and j: the higher the value of MI_{ij}, the more likely the nodes i and j are connected in the target network, and hence the greater the probability that our model will englobe interactions $i \rightarrow j$ or $i \leftarrow j$. We compute the MI matrix based on the software implementation by [15].

It is important to notice that the MI matrix may contain erroneous information, since the same is extracted from a data set that is characteristically noisy. In addition, as this matrix is symmetric, many false positive connections may be inferred when the undirected graph is transformed into a directed one: a high MI_{ij} value will enforce both $i \rightarrow j$ and $i \leftarrow j$ connections. Having said that, it is important that the method to be combined with MI uses this prior information solely as a support for its search, rather than as a picture of the true network structure. Applying random searches by means of GA over a network somehow based on MI has the benefit of guiding the exploration of the search space without restricting the stochastic nature of GA.

III. Model

As the goal of the present paper is to study the effects of an epsilon-greedy strategy to balance between random exploration and exploitation of prior knowledge in the mutation operator, we resort to the network modeling and GA implementation published by Mendoza and Bazzan [16] to test the proposed approach. In what follows we briefly explain the combination scheme between GA and RBNs, which was previously applied to the problem of reverse engineering gene regulatory networks (GRNs) [16], emphasizing some algorithmic changes introduced in the current work.

A. Representation

The structure and dynamics of GRNs were described by RBNs, presented in Section II-A. In this context, nodes represent genes and the edges depict the regulatory interactions that exist among them. Furthermore, each GA individual represents a candidate network codified into a string. In [16], authors implemented a binary encoding including the nodes' state, predictors and Boolean function. However, since we are interested in analyzing the improvements in terms of structure inference accuracy, we do not encode the nodes' states or their Boolean functions. Instead, we adopt a distinct approach and encode solely the network wiring as an integer string, which contains the set of predictors for each node in the network. An example of this representation is depicted in Fig. 1(b), in which nodes are denoted as A, B and C for the sake of simplicity.

B. Fitness function

Inspired by the work of Nam and colleagues [17], in which a sub-exhaustive search was conducted towards network topologies consistent with a temporal expression signal, Mendoza and Bazzan proposed the application of GA to explore the solutions space of consistent models. Thus, instead of entailing a search for consistent structures, the authors evolve a set of candidate solutions encoded as GA individuals and evaluate them based on an inconsistency ratio (IR) computed in relation to their network structure and the input temporal expression signals, as follows:

$$IR_i = w^{-1} \sum_{k=1}^{2^{K_i}} \min(w_k(0), w_k(1)) \tag{3}$$

in which w represents the weight of each measurement, and variables $w_k(0)$ and $w_k(1)$ denote the total weight of measurements whose output value is 0 and 1, respectively, for each $k = 1, \ldots, 2^{K_i}$ possible input combination of a node i. While the predictors' states are observed at time t, the target node's state is observed at time $t + 1$. Once the IR for each node is calculated, the network inconsistency (IR_n) is determined by the sum of all nodes' IR.

The goal of the GA is to minimize the IR_n regarding the input temporal expression signal. The fitness function is thus defined as in (4), such that individuals with lower IR are more likely to remain in subsequent generations. The term NI denotes the number of actual inferred interactions and N^2 is the maximum number of possible interactions.

$$f = \frac{1}{1 + \frac{IR_n}{0.5 \times N} + \frac{NI}{N^2}} \tag{4}$$

C. Crossover and mutation

In [16], authors applied purely random crossover and mutation operators. Given the binary strings of two individuals, crossover was performed by randomly selecting two points in the interval $[1, L]$, where L denotes the length of the strings, and swapping all the genetic material among both points between the pair of individuals. Furthermore, point mutations were applied as bit inversion, in which a set of randomly selected bits had their value flipped.

In the present paper we implement some changes over these operators following the direction discussed in [18], in which crossover and mutation were performed over network wiring. This strategy aims at finding the best network structure by varying the connections between nodes and looking for the combination that maximizes the fitness function. Therefore, crossover is implemented as a connections swap among a subset of nodes of two GA individuals. As an example, consider the situation exposed in [18]: given two individuals of length 10, referring to networks of $N = 5$

Algorithm 1 The epsilon-greedy mutation operator

```
1: for each individual in population do
2:   if random ≤ P_mut then
3:     randomly choose a pair of nodes i and j;
4:     extract the belief MI_ij from MI matrix;
5:     if random ≤ P_prior then
6:       mutate link (i, j) by exploiting prior knowledge MI_ij;
7:     else
8:       mutate link (i, j) by randomly exploring search space;
9:     end if
10:  end if
11: end for
12: P_prior = P_prior × Δ;
```

nodes and $K = 2$, `0545120300` and `2534102340`, if the random choice of the operator is to start the crossover at point 3, all connections regarding nodes 3 to N will be exchanged between the pair of mates to generate the offspring, which in this case will be `0545122340` and `2534100300`. In the next section we present our epsilon-greedy mutation operator.

IV. An Epsilon-Greedy Mutation Operator

In the current work, we propose an epsilon-greedy mutation operator to replace the traditional GA blind, random mutation. This operator is based on an epsilon-greedy strategy and aims at alternating between random operations and decisions made upon available prior knowledge. The epsilon-greedy approach has been frequently used to achieve a trade-off between exploration and exploitation in other scenarios [11] – a phenomenon that we intend to reproduce in the context of networks inference via GA.

The basic functioning of the proposed epsilon-greedy mutation operator is described in Algorithm 1. Our operator works with two probabilities: P_{mut} and P_{prior}. The trade-off between exploration and exploitation is controlled by $P_{prior} = 1 - \epsilon$, which is the probability of using prior knowledge when performing a mutation. When $\epsilon = 0$, P_{prior} will be equal to 1 and thus our mutation operator will follow an exploitative policy (Algorithm 1, line 6). In this case, the probability of performing a mutation over a randomly selected link will be determined by the prior knowledge. In the present paper, we have used a normalized MI matrix as the prior information such that the higher the belief MI_{ij} attached to a link between genes i and j ($\text{MI}_{ij} \gg 0.5$), the more likely this link will be added to our model during a mutation. Conversely, the smaller this belief ($\text{MI}_{ij} \ll 0.5$), the more probable a mutation will remove this link. In contrast, when $\epsilon = 1$, $P_{prior} = 0$ and hence our operator reproduces the traditional GA blind, random mutation (Algorithm 1, line 8).

Simulations start with $\epsilon = 0$, thus allowing GA to be highly exploitative during the first generations. This means that the networks encoded by early GA populations will be very close to the network structure inferred by MI. The probability P_{prior} is then gradually decreased throughout generations by a multiplicative factor Δ (Algorithm 1, line 12), the annealing, whose value is the parameter that controls convergence speed. As P_{prior} decreases, more random searches will be performed over the MI-based initial networks. The inferior limit for P_{prior} is zero, which refers to the case where $\epsilon = 1$ and hence, that GA will follow a purely explorative approach.

The decision about attempting a mutation over an individual follows the traditional approach: a probability P_{mut} is applied to decide whether the GA will propose to mutate the genetic material of a candidate solution. However, the actual occurrence of a mutation depends on other factors, such as the belief associated with the link to be mutated in the case of using prior knowledge, or if the operations proposed are free of redundancy and valid from the viewpoint of network syntax, i.e., they satisfy some constraints like the maximum connectivity allowed for each node.

V. Experiments and Results

We run our experiments with a benchmark network obtained with an Artificial Gene Network (AGN) validation and simulation model [19]. As our study aims at investigating the effects of an epsilon-greedy mutation operation over inference accuracy, rather than the efficiency of a specific network inference method, we test the enhancements proposed with a single deterministic 50-node target network, which is generated according to a scale-free Boolean network model. The scale-free topology [20] is currently known to be one of the most similar models to real biological networks [21], which is the study case in [16]. The upper bound limit of nodes' average connectivity was configured as $\langle k \rangle = 3$ [22]. We simulated 10 temporal expression signals of length 10, each one starting from a randomly chosen initial state and concatenated them into a single time series used for network inference.

GA parameters were configured as described in [16], except for P_{mut}. We evolved a population of 50 individuals over 1000 generations, applying crossover with probability $P_{cross} = 1$ and elitism with an elite size $E = 4$. The probability of attempting a mutation P_{mut} was varied between 0.1 and 0.5 in steps of 0.1. We tested and analyzed the effects of the proposed epsilon-greedy mutation operator applying the annealing factors $\Delta = \{0.9, 0.99, 0.999\}$ over P_{prior}. We compare this results with the two extreme situations, i.e., a purely exploitative ($\epsilon = 0$, no decay) and a purely explorative ($\epsilon = 1$, no decay) GA algorithm, where the latter mimics the traditional blind, random GA mutation. For each of these scenarios, we performed 30 simulations and build an ensemble prediction from each of which by letting the best individuals of the final generation to perform a majority voting on the network structure. Furthermore, we combine all ensemble predictions into a single consensus network and use it for results assessment in the current work.

Table I
EFFECTIVE MUTATION RATES FOR SIMULATIONS WITH NO DECAY.

	P_{mut}				
	0.1	0.2	0.3	0.4	0.5
$\epsilon = 0$, no decay	0.32%	0.61%	0.91%	1.17%	1.49%
$\epsilon = 1$, no decay	3.47%	5.81%	8.05%	9.61%	11.39%

The effective mutation rates for simulations with no P_{prior} decay appear in Table I. Readers may notice that albeit P_{mut} is configured with relatively large values considering traditional GA setups, the number of actual mutations performed is lower due to the set of conditions that must be satisfied in order to a mutation in fact occur. When a decay rate Δ is applied, the effective mutation rates range from the values observed in the extreme cases, i.e, $\epsilon = 0$ and $\epsilon = 1$, being very close to the latter when $\Delta = 0.9$.

A comparison in terms of the average fitness among the scenarios tested is shown in Fig. 3. The graphs behavior evidences the better results obtained when prior knowledge is applied ($\epsilon = 0$), especially for no decay and decay rate $\Delta = 0.999$: the average fitness after 1000 generations is higher than values for the traditional GA. Furthermore, the use of prior knowledge combined to low decay rates seems to originate populations with greater internal variability. This improvement comes at a cost though: the convergence speed for these cases is slightly slower than for the remainder. As this tendency was observed for all mutation rates, we give here solely the results for $P_{mut} = 0.3$. A trade-off between performance and speed can be achieve by tuning the method's parameters, such as the decay rate Δ.

Furthermore, we compare the results based on the AUC score of the inferred networks, which is computed as the area under the ROC curve (Table II). The highest scores for each P_{mut} value are emphasized in boldface style. Except for $P_{mut} = 0.4$, in which the purely random exploration approach ($\epsilon = 1$) has yielded the best results, the epsilon greedy mutation operator has introduced improvements up

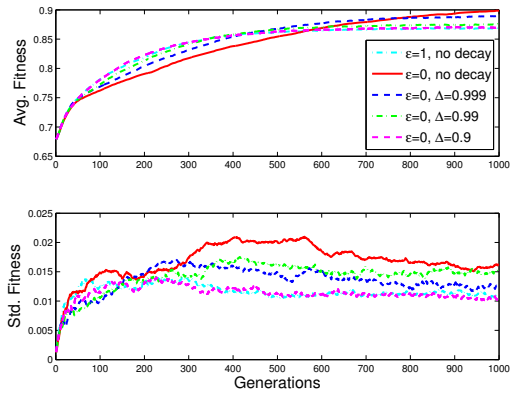

Figure 3. Evolution of average fitness values when $P_{mut} = 0.3$.

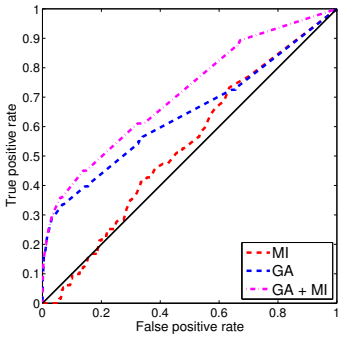

Figure 4. A comparison between the performance of the proposed coupling scheme (GA + MI) and the individual methods when $P_{mut} = 0.3$.

to 11% higher than the traditional GA, providing the most accurate inferences in most of the scenarios.

To stress even more the benefits of combining GA and MI via our epsilon-greedy mutation operator, we have assessed the network inferred solely from the MI matrix as described in Section II-B, using as threshold every MI value in the matrix in an ascending order, and have found an AUC score of 0.5345. This means that the coupling scheme between GA and MI embedded in the proposed epsilon-greedy mutation operator outperforms both methods when individually applied, as shown in Fig. 4. Again, only the $P_{mut} = 0.3$ case is depicted as this behavior is general. However, there is no consensus regarding the best annealing value (Δ), since enhancements were observed for all P_{prior} decay rates.

VI. CONCLUSION

In the present paper we introduced an epsilon-greedy mutation operator as a means of improving accuracy of networks inference by GA through the use of prior knowledge. As far as we are concerned, up to now, this strategy has not been applied to this purpose. Our operator balances between random mutations and mutations decided upon some available prior knowledge, which in the scope of this work was obtained from a MI matrix. Experiments have shown that the proposed coupling scheme provides a faster and better convergence in relation to the traditional GA. In addition, the AUC scores of inferred networks were boosted, achieving superior marks than those provided by both methods when individually applied. Furthermore, this mutation operator allows user to control GA evolvability and the trade-off between exploration and exploitation by adjusting the rate with which the probability of using prior knowledge is decreased. For future works, we intend to test this approach with distinct prior knowledge sources and compare the respective sensibility and performance.

ACKNOWLEDGMENT

We thank the Brazilian National Council for Research (CNPq) for its support to the authors.

Table II
RESULTS IN TERMS OF AUC SCORE OBTAINED FROM THE CONSENSUS NETWORK BUILT AFTER 30 SIMULATIONS.

P_{mut}	$\epsilon = 0$, no decay	$\epsilon = 1$, no decay	$\epsilon = 0, \Delta = 0.999$	$\epsilon = 0, \Delta = 0.99$	$\epsilon = 0, \Delta = 0.9$
0.1	0.5848	0.6567	0.6474	**0.6738**	0.6624
0.2	0.6047	0.6667	0.6793	**0.7034**	0.6764
0.3	0.5964	0.6414	0.6487	0.6979	**0.7133**
0.4	0.5823	**0.6726**	0.6482	0.6538	0.6200
0.5	0.5542	0.6568	**0.6955**	0.6847	0.6925

REFERENCES

[1] J. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor: Univ. of Michigan Press, 1975.

[2] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[3] F. Leung, H. Lam, S. Ling, and P. Tam, "Tuning of the structure and parameters of a neural network using an improved genetic algorithm," *Neural Networks, IEEE Transactions on*, vol. 14, no. 1, pp. 79–88, jan 2003.

[4] F. H. van Batenburg, A. P. Gultyaev, and C. W. Pleij, "An APL-programmed genetic algorithm for the prediction of RNA secondary structure." *J Theor Biol*, vol. 174, no. 3, pp. 269–280, Jun. 1995.

[5] C. Cotta and J. M. Troya, "Reverse engineering of temporal Boolean networks from noisy data using evolutionary algorithms," *Neurocomputing*, vol. 62, pp. 111–129, 2004.

[6] A. Carvalho, "A cooperative coevolutionary genetic algorithm for learning Bayesian network structures," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, ser. GECCO '11. New York, NY, USA: ACM, 2011, pp. 1131–1138.

[7] K. Deep and M. Thakur, "A new mutation operator for real coded genetic algorithms," *Applied Mathematics and Computation*, vol. 193, no. 1, pp. 211–230, 2007.

[8] D. Adler, "Genetic algorithms and simulated annealing: a marriage proposal," in *IEEE International Conference on Neural Networks*, vol. 2, 1993, pp. 1104–1109.

[9] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994.

[10] Y. Sasaki and H. de Garis, "Faster evolution and evolvability control of genetic algorithms using a softmax mutation method," in *Proceedings of the 2003 Congress on Evolutionary Computation CEC2003*, R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, Eds. Canberra: IEEE Press, 2003, pp. 886–891.

[11] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Machine Learning: ECML 2005*, ser. Lecture Notes in Computer Science, J. Gama, R. Camacho, P. Brazdil, A. Jorge, and L. Torgo, Eds. Springer Berlin / Heidelberg, 2005, vol. 3720, pp. 437–448.

[12] 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*, vol. 3, 1998, pp. 18–29.

[13] A. J. Butte, I. S. Kohane, and I. S. Kohane, "Mutual information relevance networks: Functional genomic clustering using pairwise entropy measurements," *Pacific Symposium on Biocomputing*, vol. 5, pp. 415–426, 2000.

[14] A. Margolin, K. N. Basso, C. Wiggins, G. Stolovitzky, R. Favera, and A. Califano, "ARACNE: An algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context," *BMC Bioinformatics*, vol. 7, no. Suppl 1, p. S7, 2006.

[15] P. Qiu, A. J. Gentles, and S. K. Plevritis, "Fast calculation of pairwise mutual information for gene regulatory network reconstruction," *Computer Methods and Programs in Biomedicine*, vol. 94, no. 2, pp. 177–180, 2009.

[16] M. R. Mendoza and A. L. C. Bazzan, "Evolving Random Boolean Networks with Genetic Algorithms for Regulatory Networks Reconstruction," in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '11. New York, NY, USA: ACM, July 2011, pp. 291–298.

[17] D. Nam, S. Seo, and S. Kim, "An efficient top-down search algorithm for learning boolean networks of gene expression," in *Machine Learning*. Springer Science, 2006, vol. 65, pp. 229–245.

[18] M. R. Mendoza, F. M. Lopes, and A. L. C. Bazzan, "Reverse Engineering of GRNs: an Evolutionary Approach based on the Tsallis Entropy," in *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, 2012.

[19] F. M. Lopes, R. M. Cesar-Jr, and L. da F. Costa, " AGN Simulation and Validation model," in *Proceedings of Advances in Bioinformatics and Computational Biology*, vol. 5167 of Lecture Notes in Bioinformatics. Springer-Verlag Berlin, 2008, pp. 169–173.

[20] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[21] R. Albert, "Scale-free networks in cell biology," *J Cell Sci*, vol. 118, no. 21, pp. 4947–4957, 2005.

[22] S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets." *J Theor Biol*, vol. 22, no. 3, pp. 437–467, March 1969.