ISSN 1817-3721, E-ISSN 1818-8745

Plant Tissue Cult. & Biotech. **28**(2): 237-250, 2018 (December)
©Bangladesh Assoc. for Plant Tissue Culture & Biotechnology

PTC&B

Inferring Gene Regulatory Network with Recurrent Neural Network and Extended Artificial Bee Colony Algorithm

Tausif Al Hossain, Mohammad Shoyaib[1] and Saifuddin Md. Tareeq*

Department of Computer Science and Engineering, University of Dhaka, Dhaka-1000, Bangladesh

Key words: Gene regulatory network, Recurrent neural network, Artificial bee colony

Abstract

Gene regulatory network is the network of genes interacting with each other performing as functional circuitry inside a cell. Many cellular processes are controlled by this network as they govern the expression levels of genes or gene product. High performance computational techniques are needed to analyze these data as it is heavily affected by noise. There are a number of algorithms available in the literature which use recurrent neural network for model building together with differential evolution, particle swarm optimization or genetic algorithm for searching the regulatory network. The problem with these methods is that they may trap in the local minimum. In this paper, we present an algorithm using recurrent neural network as model and an extended artificial bee colony algorithm for searching regulatory network that can avoid local minimum. A comprehensive analysis on both artificial and real data shows the effectiveness of the proposed approach. Furthermore we have also varied the network dimension and the noise level present in gene expression profiles. The reconstruction method has successfully predicted the underlying network topology while maintaining high accuracy. The proposed approach has also been applied to the real expression data of SOS DNA repair system in *Escherichia coli* and successfully predicted important regulations.

∗Author for correspondence: <smtareeq@cse.du.ac.bd>. [1]Institute of Information Technology, University of Dhaka, Dhaka-1000, Bangladesh.

DOI: https://doi.org/10.3329/ptcb.v28i2.39682

Introduction

Gene expression levels are the outcome of regulatory interactions among genes. Therefore, gene regulatory network provides information about the functionality inside a cell. DNA microarray technology provides the way to measure the expression levels of thousands of genes simultaneously. These expression levels are the outcome of regulatory relations among genes. Therefore, inferring gene regulatory network from these microarray data is known as a reverse engineering problem. Though it seems like a traditional inverse problem, the solution is not trivial. It presents a large number of unknowns within a small sample size. Further, DNA microarray data are affected by noise. As a result, mathematical modeling techniques with reliable inference algorithm are needed to solve the problem.

Different techniques have been developed to solve the gene regulatory network inference problem. Two main components are involved with this problem; a "mathematical model" that describes the relations among genes and a "search strategy" to find related parameters of the mathematical model such that we can build a network. Boolean network (Maria and Stefan 2008) is computationally expensive to analyze larger networks (Guy and Shamir 2008), complexity of Bayesian network (Friedman et al. 2000, Perrin et al. 2003) is exponential (Savageau 1976, Noman and Iba 2007) and perform well only in small and medium sized network. A linear time varying model based approach has been proposed by Kabir et al. 2010. With this model a self-adaptive differential evolution approach is used as reverse engineering algorithm. A reverse engineering approach based on the recurrent neural network (RNN) formalism has been proposed by Xu et al. 2004, Noman et al. 2013, Xu et al. 2007, Akbarzadeh et al. 2011, Hu et al. 2006. Most RNNs have had scaling issues. In particular, RNNs cannot be easily trained for large number of neuron units nor for large number of inputs units.

Different algorithms have been used so far to address the problem of searching gene regulatory network. Most of them are evolutionary or swarm based approaches. Differential evolution (Storn and Price 1997, Noman and Iba 2007, Xu et al. 2007, Noman and Iba 2006, Chowdhury and Chetty 2011) iteratively tried to improve a candidate solution based on a given measure of quality. Particle Swarm Optimization (PSO) first proposed by Kennedy and Eberhart 1995 and different approaches (Xu et al. 2004, Xu et al. 2007) of PSO have been used as the inference algorithm. PSO starts with random candidate solutions or particles. Then it iterates and changes the velocity of each particle to the better solution.

The most commonly used fitness evaluation criterion is the difference between the gene expression levels calculated and the observed system dynamics. It is known as the mean squared error (MSE) based fitness function (Noman et al. 2013, Chowdhury and Chetty 2012). Another type of fitness function is based on information criteria. Information criteria provide a simple method to choose from a range of competing models. AIC (Akaike 1998) is the most commonly used information criterion. AIC based fitness function was proposed by Noman and Iba 2007.

In this paper an algorithm based on recurrent neural network framework with extended artificial bee colony algorithm is proposed. The inference algorithm incorporates mutation and crossover operators from genetic algorithm to avoid local minimum where some other algorithms may be stuck.

Materials and Methods

First a small 4 gene artificial network is used to test the proposed approach. This network is studied before by Noman et al. 2013. In this case, we have used noiseless, 5% noisy, 10% noisy and 20% noisy data. This noise adding approach is $\tilde{x}_i(t) = x_i(t)\times(1+\eta), (-R \le \eta \le R)$ where η represents the percentage of Gaussian noise.

Multiple time points and multiple data sets have also been used to test the algorithm. All the setup conditions and the parameters associated with the network are given in Table 1.

Table 1. Parameter associated with 4 genes artificial network.

w_{ij}				β_i	τ_i
20.0	–20.0	0.0	0.0	0.0	10.0
15.0	–10.0	0.0	0.0	–5.0	5.0
0.0	–8.0	12.0	0.0	0.0	5.0
0.0	0.0	8.0	–12.0	0.0	5.0

Ten data sets each containing 10 time points for each gene have been generated to test the algorithm. For the second case we have used 30 genes network and the same network used by Noman et al. 2013. We have used noiseless and 5% noisy data in this case.

For the real data, the proposed algorithm is tested to reconstruct the SOS DNA repair network in *Escherichia coli*. Gene expression data set collected by Uri Alon Lab was used in the experiment (Alon 2015). This data set contains expression levels of 8 genes and expression levels were measured after irradiation of DNA with UV light. Four experiments were carried out with different light intensities. Each experiment collected 50 sample points at 6 min interval. Here we have used the first data set only. The sample point at first time stamp was ignored as it was zero. Also the sample point values were normalized in the range (0,1).

There are two performance metrics associated with this inference problem namely sensitivity and specificity and defined in equations (1) and (2), respectively.

$$S_p = \frac{TP}{TP + NP} \qquad (1)$$

$$S_n = \frac{TN}{TN + FP} \qquad (2)$$

Here TP = number of true positive, FP = number of false positive, TN = number of true negative, FN = number of false negative.

A recurrent neural network (RNN) is a special type of neural network where connections between units form a directed cycle. Fig. 1 shows a simple recurrent neural network model. Equation (3) expresses the RNN model (Dhaeseleer et al. 2000).

$$\tau_i \frac{de_i}{dt} = f\left(\sum_{j=1}^{N} \omega_{ij} e_j + \sum_{k=1}^{K} v_{ik} u_k + \beta_i\right) - \lambda_i e_i \qquad (3)$$

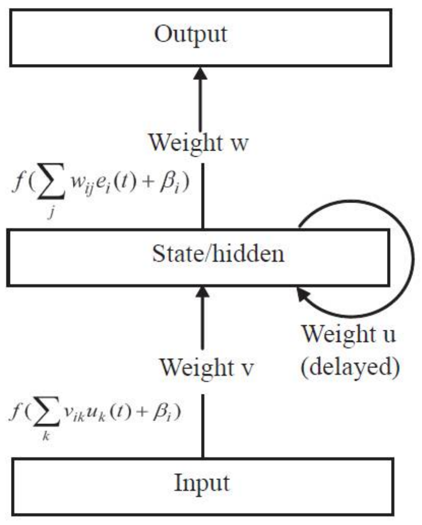
Output
Weight w
$f(\sum_j w_{ij} e_i(t) + \beta_i)$
State/hidden
Weight u (delayed)
Weight v
$f(\sum_k v_{ik} u_k(t) + \beta_i)$
Input

Fig. 1. A simple recurrent neural network.

where, e_i is the expression level of ith gene, f is a nonlinear function (usually a sigmoid function is used $f(z) = 1/(1+ e^{-z})$, w_{ij} is the effect of jth gene on ith gene, K is the number of external variables, u_k is the kth ($1 \le k \le K$) external variable, v_{ik} is the effect of kth external variable on ith gene, β is the bias term, τ is the time constant, λ is the decay rate parameter. The above network is unfolded in time from $t = 0$ to T with an interval Δt. Here, the regulatory network is shown as fully connected; although in practice, the network is sparse. This model is described in a discrete form in equation (4).

$$e_i(t+\Delta t) = \frac{\Delta t}{\tau_i} f\left(\sum_{j=1}^{n} w_{ij} e_j(t) + \sum_{k=1}^{K} v_{ik} u_k(t) + \beta_i\right) + \left(1 - \frac{\lambda_i \Delta t}{\tau_i}\right) e_i(t) \qquad (4)$$

As it is difficult to obtain measurements of external variables, it is a common practice to ignore the $v_{ik}u_k$ term. Also for simplicity the decay rate parameter λ is assumed to be 1. So the simplified model equation (5) represents the final model where we have to find w, β and τ.

$$e_i(t+\Delta t) = \frac{\Delta t}{\tau_i} f\left(\sum_{j=1}^{n} w_{ij} e_j(t) + \beta_i\right) + \left(1 - \frac{\lambda_i \Delta t}{\tau_i}\right) e_i(t) \qquad (5)$$

Artificial Bee Colony (ABC) algorithms (Karaboga 2005), are very effective to optimization problems (Drias et al. 2005, Lucic and Teodorovic 2001, Quijano and Passino 2007). ABC algorithm can be used for optimizing functions with multiple variables and numerical optimization (Domnguez 2009, Karaboga and Basturk 2007, Karaboga 2009, Omkar et al. 2011). Also it can be used to train neural networks (Karaboga et al. 2007, Karaboga and Ozturk 2009).

In this paper, an extended ABC algorithm has been developed to infer gene regulatory network based on recurrent neural network model. The strategy is described in Algorithm 1. After each W interval the lowest K percent (based on fitness value) solutions are abandoned and reinitialized with new random solutions.

The MSE based fitness function for N genes using single set of time series data containing T time points is given by equation (6).

$$Fit(x_i) = \frac{1}{TN} \sum_{t=0}^{T} \sum_{i=1}^{N} (e_i(t) - d_i(t))^2 \qquad (6)$$

here, $e(t)$ is the generated time series from solution x_i and $d(t)$ is the expected time series. Multiple set of time series data has been used in this approach. So for Q number of data sets the fitness function is given in equation 7.

$$Fit(x_i) = \frac{1}{QTN} \sum_{q=1}^{Q} \sum_{t=0}^{T} \sum_{i=1}^{N} (e_i(t) - d_i(t))^2 \quad (7)$$

Algorithm 1. Extended artificial bee colony algorithm.

```
1   Initialize P employed bees using RANDOM-INITIALIZATION procedure
2   G = 1
3   repeat
4       for i ← 1 to P do
5           Produce new solution $y_i$ from $x_i$ using MUTATION and CROSSOVER
            procedure
6           Evaluate fitness of the new solution $y_i$
7           Replace $x_i$ with $y_i$ if $y_i$ has better fitness
8       end
9       for i ← 1 to P do
10          Choose a solution $x_i$ whose fitness is in higher K percent among all
            solutions
11          Produce new solution $y_i$ from $x_i$ using MUTATION and CROSSOVER
            procedure
12          Evaluate fitness of the new solution
13          Replace $x_i$ with $y_i$ if $y_i$ has better fitness
14      end
15      Choose solution with base fitness as $x_{best}$ from all solutions $x_i$ in P
16      Perform SCOUT-PHASE procedure after each W interval
17      G = G+1
18  Until G = $G_{max}$
```

Procedure RANDOM-INITIALIZATION

1 Randomly select x% employed bees from population P
2 Initialize regulatory parameters w_{ij} of those employed bees with 0
3 Initialize bias terms β and time constants τ of those employed bees randomly
4 Randomly initialize all parameters w_{ij}, β_i, τ_i of the remaining employed bees

Procedure MUTATION

1 For each individual x_G^i in current generation, a mutated individual y_i is generated by the equation:

$$y_G^i = x_G^j + F(x_G^k - x_G^l)$$

where *i* # *j* # *k* = *l*. F is the scaling factor.

Procedure CROSSOVER

1 The mutated individual y_G^i has a crossover with x_G^i and generates y_{G+1}^i. .
2 In this crossover parameters of the offspring are inherited from x_G^i or y_G^i

depending on the value of crossover factor CF, i.e if $r \leq CF$ then it is inherited from x_G^i otherwise from y_G^i.

```
Procedure SCOUT-PHASE
1  Find best solution $f_{best}$ from onlooker bees
2  Find worst solution $f_{worst}$ from onlooker bees
3  if $f_{worst} - f_{best} < \delta_{worst,best}$
4      Select lowest M percent of employed bees based on fitness
5      Randomly initialize those selected employed bees with new candidate solutions
6  end
```

Results and Discussion

To confirm the effectiveness of the proposed algorithm first it is applied to artificial noiseless and noisy data and then to real data. The algorithm is also tested on smaller artificial network. Then a larger artificial network is used to test scalability of the solution. For artificial data both noiseless and noisy data is used. Noisy data was used because practically microarray data contain noise.

Initial population size is considered 100, range of regulatory parameter is –30.0 to +30.0 initialized to 0, range of bias term is –10.0 to +10.0 initialized randomly and time constant is from 1.0 to 20.0 initialized randomly. First a small 4 gene network is used to test the proposed approach using noiseless data first and then by adding noise (5, 10 and 20%).

Noise free data: First we have tested the algorithm on noise free data. Inferred parameters from noise free data are given in Table 2. We can see all the parameters have been identified correctly. There are no false positives or false negatives. The values are very much precise and exact. Inferred network is as same as the original network. The algorithm successfully reverse engineered the input network. Values of sensitivity and specificity are given in Table 2.

5% noisy data: Result shows that for 5% noise the proposed approach performed well. All the regulations have been identified correctly. One non regulation is missed. Thus it generates a false positive. But no false negative is generated. In some cases inferred parameter values are degraded. But overall the network structure is identified with good efficiency. Values of sensitivity and specificity are given in Table 2. Sensitivity value is same as before. But as there is a false positive generated specificity is slightly reduced. Overall the algorithm performed well.

10% noisy data: Inferred parameters for 10% noisy data are shown in Table 2. This time also the algorithm identified all the regulations successfully. Among the non-regulations it missed two and assigned them as regulations. Therefore, two false positives are generated. But there is no false negative. Therefore the network structure is identified with some false edges. Parameter values are degraded more. In some cases they have a large margin from the original values. Specially, the bias terms and time constants are not very precise. Values of sensitivity and specificity are given in Table 2. Sensitivity value is as same as before. Specificity value reduces further due to the increasing number of false positives.

20% noisy data: Inferred parameters for 20% noisy data are given in Table 2. This time six regulations have been identified correctly. One false negative is generated this time too. Two non-regulations have been missed. And one regulation has wrong sign. Therefore, total three false positives have been generated. Inferred parameter values are not very specific. Sensitivity and specificity values are given in Table 2. Sensitivity value is reduced for the first time due to the presence of the false negative and specificity value is further degraded due to the increasing number of false positive regulations. Reconstructed gene regulatory network for 4 genes artificial data is shown in Fig. 2.

Table 2. Parameter, sensitivity and specificity of 4 gene artificial network

	w_{ij}				β_i	τ_i	S_n	S_p
Noiseless	20.01	–20.00	0.00	0.00	0.00	10.17		
	15.07	–10.00	0.00	0.00	–4.98	5.10	1.00	1.00
	0.00	–8.07	12.06	0.00	0.00	5.00		
	0.00	0.00	8.86	–12.00	0.00	5.00		
5% noisy	22.72	–20.56	0.00	0.00	0.07	10.11		
	15.87	–11.07	–2.20	0.00	–5.20	6.02	1.00	0.86
	0.00	–8.25	12.33	0.00	0.77	5.78		
	0.00	0.00	8.78	–12.78	0.05	5.36		
10% noisy	23.33	–21.29	0.00	0.00	0.12	11.11		
	16.76	–11.93	0.00	0.00	–6.22	5.51	1.00	0.86
	5.04	–8.01	13.76	0.00	0.88	5.79		
	0.00	0.00	9.21	–12.91	3.21	6.79		
20% noisy	30.00	–26.66	0.00	0.00	5.21	11.54		
	18.11	–15.51	8.86	0.00	6.32	10.42	0.86	0.75
	0.00	0.00	16.88	0.00	–7.77	8.81		
	-9.72	0.00	18.80	–15.12	8.20	11.10		

30 gene artificial network: Next experiment is done with a larger network of 30 genes. Network structure is sparse and it is the same network used by Noman et al. 2013. Among the possible 900 connections only 36 are regulations and rest are non regulations. So the graph is very sparse. Experiments were done using both noiseless and noisy data. For noisy data 5% noise was added to the data sets. Experimental setup and environment were same as 4 gene network.

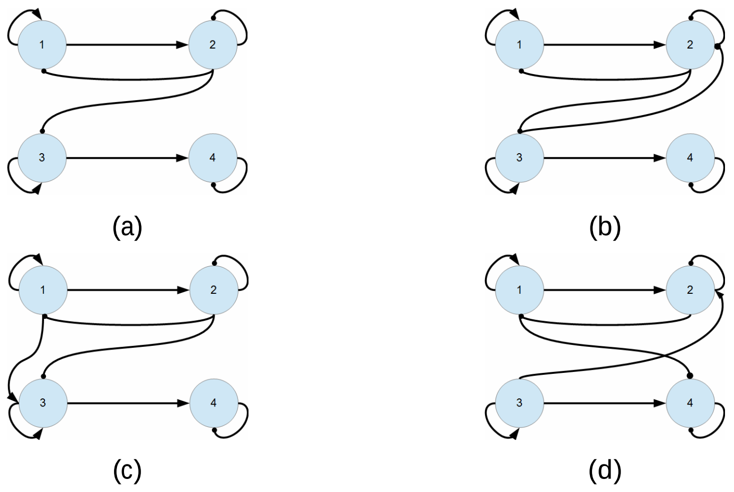

Fig. 2. Reconstructed 4 gene regulatory network with (a) noiseless, (b) 5% noisy, (c) 10% noisy and (d) 20% noisy data.

With the increase of dimensions the problem complexity increases rapidly. In noise free condition the algorithm identified almost 70% regulations correctly. Most of the non regulations also identified correctly. But at the same time the algorithm predicted some false positives and false negatives. For 5% noisy data the identified almost 55% regulations correctly. Almost 85% non regulations have been identified correctly. But again the number of false positives and false negatives increased. Overall performance of the algorithm are given in Table 3. The table shows that performance of the algorithm degrades with the increment of noise level present. However it can be used for larger and sparse networks given sufficient amount of gene expression data.

Table 3. Sensitivity and specificity of 30 gene artificial network.

	Noisy free data	5% noisy data
Sensitivity, S_n	0.78	0.67
Specificity, S_p	0.92	0.86

Experiment with real data: The proposed algorithm was tested to reconstruct the SOS DNA repair network in *Escherichia coli* shown in Fig. 3. This network consists of 40 genes. The network is initiated when any damage is detected in DNA.

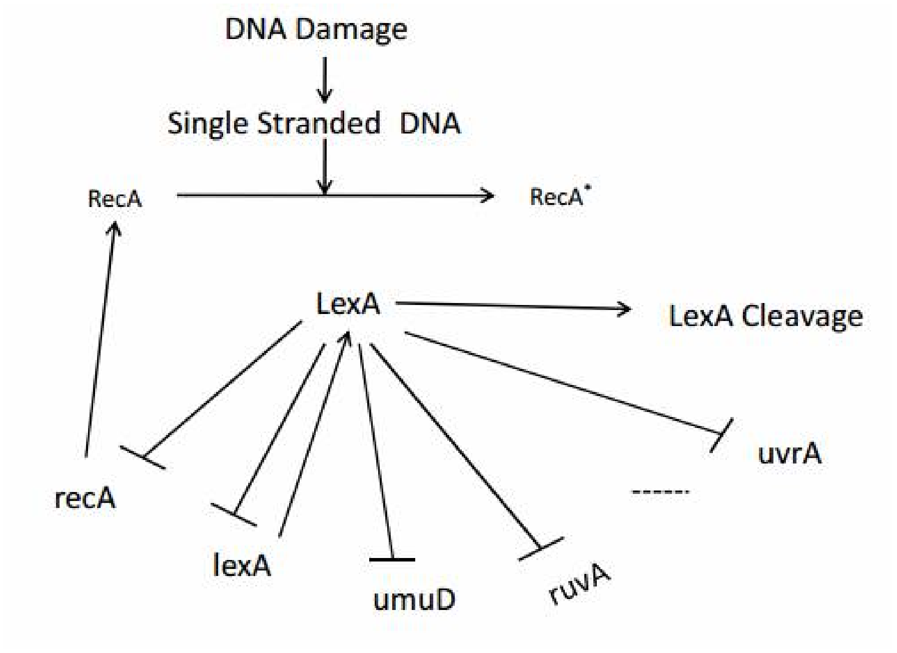

Fig. 3. SOS DNA repair network in *Escherichia coli*.

Protein *RecA* gets activated when DNA damage takes place. It mediates *LexA* by binding to single stranded DNA molecules. Protein *LexA* is a master repressor which represses all other genes when no damage takes place. Mediated level of *LexA* starts activation of other genes. After repairing the damage *RecA* expression level drops. Then *LexA* binds site in promoter region of SOS genes, repress their expression levels and restores original states.

Experimental setup is kept same as before. Experiment was repeated for 10 individual trials. In each trial a very small fitness score was achieved which indicated that the proposed algorithm could predict a network that match the target time series data pretty well. Predicted regulations from the experiment are shown in Table 4.

From the result it is observed that the proposed algorithm has identified a number of regulations. Inhibition of *lexA* on all other genes, also the activation of *recA* on *uvrY* and *polB* has been identified correctly. The prediction contains some false positives which are either unknown regulations or the side effect of noise present in the data. Results are compared with other methods in terms of truly identified regulations. Comparison shows the competitiveness of the proposed approach. Comparative results are shown in Fig. 4.

Table 4. Estimated regulations for SOS DNA repair system.

Gene	Predicted regulation	References
uvrD	uvrD ┤ uvrD, lexA ┤ uvrD	(Kimura et al. 2008, Cho et al. 2006, Kimura et al. 2009)
lexA	uvrD → lexA	(Kimura et al. 2008, Cho et al. 2006)
umuD	lexA ┤umuD, uvrY → umuD	(Kimura et al. 2009, Gardner et al. 2003)
recA	uvrD → recA, lexA ┤recA, ruvA ┤recA	(Kimura et al. 2008)
uvrA	umuD → uvrA, uvrY → uvrA	(Perrin et al. 2003, Kabir et al. 2010, Kimura et al. 2009)
uvrY	lexA → polB, uvrA ┤ polB	(Kabir et al. 2010, Kimura et al. 2009, Noman et al. 2013)
ruvA	lexA → polB, uvrA ┤ polB	(Kabir et al. 2010, Kimura et al. 2009, Noman et al. 2013)
polB	lexA ┤ polB, recA → polB	(Kabir et al. 2010, Kimura et al. 2009)

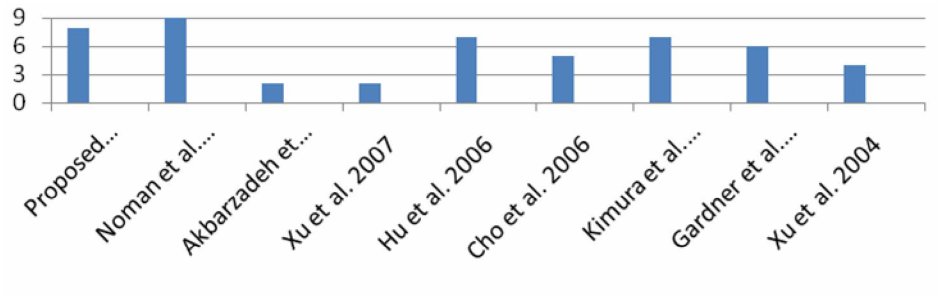

Fig. 4. Comparison of results with other methods.

Conclusions

In this work a new approach is proposed to infer gene regulatory networks. The approach is tested on artificial and real networks. Two artificial networks of 4 genes and 30 genes were used. For both the networks, the proposed algorithm performed very well. For noiseless environment, the proposed approach predicted the network with 100% accuracy. However with the increase of noise level present in the data the algorithm's accuracy has been reduced. But still its performance remained acceptable. For real data the proposed method predicted eight true regulations which is much better than most of the existing state of the art methods. Thus, along with some improvements in future this work may be served as an initiative for the future researchers in this area. In future it can be tested on large networks of thousand or more genes to evaluate performance on real large networks. To reconstruct large scale gene networks of complex organisms, an improvement in methodology, algorithmic efficiency, computing power and additional domain knowledge will also be included with the current form.

References

Akaike H (1998) Information theory and an extension of the maximum likelihood principle. Springer Series in Statistics, 199-213.

Akbarzadeh T M R, Ghazikhani A and **Monse R** (2011) Genetic regulatory network inference using recurrent neural networks trained by a multi agent system. *In:* 1st International eConference on Computer and Knowledge Engineering (ICCKE), 95-99.

Alon A (2015) Uri Alon Lab. In http://www.weizmann.ac.il/mcb/UriAlon/, last accessed on March 2015.

Cho D, Cho K and **Zhang B** (2006) Identification of biochemical networks by S-tree based genetic programming. Bioinformatics **22**: 1631-1640.

Chowdhury A R and **Chetty M** (2011) An improved method to infer gene regulatory network using s-system. Evolutionary Computation 1012-1019.

Chowdhury AR, Chetty M, **and Nguyen XV** (2012) Adaptive regulatory genes cardinality for reconstructing genetic networks. Evolutionary Computation 1-8.

Dhaeseleer P, Liang S and **Somogyi R** (2000) Genetic network inference: from coexpression clustering to reverse engineering. Bioinformatics **16**(8):707-726.

Domnguez O C (2009) An adaptation of the scout bee behavior in the Artificial Bee Colony algorithm to solve constrained optimization problems. Master's thesis, Laboratorio Nacional de Informtica Avanzada (LANIA).

Drias H, Sadeg S and **Yahi S** (2005) Cooperative bees swarm for solving the maximum weighted satisfiability problem. Computational Intelligence and Bioinspired Systems **3512**: 318-325.

Friedman N, Linial M, Nachman I and **Pe'er D** (2000) Using Bayesian networks to analyze expression data. Journal of Computational Biology **7**(3-4): 601-620.

Gardner T, di Bernardo D, Lorenz D and **Collins J** (2003). Inferring Genetic Networks and identifying compound mode of action via expression profiling. Science **301**: 102-105.

Guy K and **Ron S** (2008) Modeling and analysis of gene regulatory networks. Nature Reviews Molecular Cell Biology **9**: 770-780.

Hu X, Maglia A and **Wunsch DC** (2006) A general recurrent neural network approach to model genetic regulatory networks. *In:* 27th Annual International Conference of Engineering in Medicine and Biology Society 4735-4738.

Kabir M, Noman N and **Iba H** (2010) Reverse engineering gene regulatory network from microarray data using linear time-variant model. BMC Bioinformatics **11**(Suppl 1): S56.

Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical report, Erciyes University, Engineering Faculty, Computer Engineering Department 2005.

Karaboga N (2009) A new design method based on artificial bee colony algorithm for digital IIR filters. Journal of the Franklin Institute **346**(4):328-348.

Karaboga D, Akay B and **Ozturk C** (2007) Artificial bee colony (abc) optimization algorithm for training feed-forward neural networks. Lecture Notes in Computer Science **4617**: 318-329.

Karaboga D and **Basturk B** (2007) Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. Lecture Notes in Computer Science **4529**: 789-798.

Karaboga D and **Ozturk C** (2009) Neural networks training by artificial bee colony algorithm on pattern classification. International Journal on Neural and Mass – Parallel Computing and Information Systems **19**(3): 279 - 292.

Kennedy J and **Eberhart R** (1995) Particle swarm optimization. IEEE International Conference on Neural Networks **4**: 1942-1948.

Kimura S, Katsuki S, Soichiro Y, Hideki M, Koki M and **Mariko H** (2008) Function approximation approach to the inference of reduced ngnet models of genetic networks. BMC Bioinformatics **9**: 111-124.

Kimura S, Satoshi N and **Mariko H** (2009) Genetic network inference as a series of discrimination tasks. Bioinformatics **25**: 918-925.

Lucic P and **Teodorovic D** (2001) Bee system: modeling combinatorial optimization transportation engineering problems by swarm intelligence. Triennial Symposium on Transportation Analysis 441-445.

Maria ID and **Stefan B** (2008) Boolean network model predicts cell cycle sequence of fission yeast. PLoS ONE **3**(2): e1672, 02 2008.

Noman N and **Iba H** (2006) On the reconstruction of gene regulatory networks from noisy expression profiles. Evolutionary Computation 2543-2550.

Noman N and **Iba H** (2007) Inferring gene regulatory networks using differential evolution with local search heuristics. Computational Biology and Bioinformatics **4**(4): 634-647.

Noman N, Palafox L and **Iba H** (2013) Reconstruction of gene regulatory networks from gene expression data using decoupled recurrent neural network model. Proceedings in Information and Communications Technology **6**: 93-103.

Omkar SN, Senthilnath J, Khandelwal R, Naik GN and **Gopalakrishnan S** (2011) Artificial bee colony (abc) for multi-objective design optimization of composite structures. Applied Soft Computing **11**(1):489 -499.

Perrin B, Ralaivola L, Mazurie A, Bottani S, Mallet J and **d'AlcheBuc** (2003) Gene networks inference using dynamic bayesian networks. Bioinformatics **9**: 138-148.

Quijano N and **Passino K M** (2007) Honey bee social foraging algorithms for resource allocation. Algorithm and theory ACC '07, 3383-3388.

Savageau AM (1976) Biochemical systems analysis. A study of function and design in molecular biology. Addison-Wesley.

Storn R and **Price K** (1997) Differential evolution a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization **11**(4): 341-359.

Xu R, Hu X and **Wunsch D C** (2004) Inference of genetic regulatory networks with recurrent neural network models. IEMBS 2004, 26th Annual International Conference of the IEEE **2:** 2905-2908.

Xu R, Wunsch DC and **Frank RL** (2007) Inference of genetic regulatory networks with recurrent neural network models using particle swarm optimization. Computational Biology and Bioinformatics **4**(4): 681-692.