A Recurrent Fuzzy Neural Model of a Gene Regulatory Network for Knowledge Extraction Using Artificial Bee Colony Optimization Algorithm

Papia Das[1], Pratyusha Rakshit[2], Amit Konar[2], R.Janarthanan[3]

[1]*CSE Dept., Jadavpur University, Kolkata-700032*

[2]*ETCE Dept., Jadavpur University, Kolkata-700032*

[3]*Jaya Engineering College, Chennai*

ju.papia@gmail.com, pratyushar1@gmail.com, konaramit@yahoo.co.in, srmjana_73@yahoo.com

Abstract — Generating inferences from a gene regulatory network is important to understand the fundamental cellular processes, involving gene functions, and their relations. The availability of time-series gene expression data makes it possible to investigate the gene activities of the whole genomes. Under this framework, gene interaction is explained through a connection weight matrix. Based on the fact that the measured time points are limited and the assumption that the genetic networks are usually sparsely connected, we present an ABC-based search algorithm to unveil potential genetic network constructions that fit well with the time-series data and explore possible gene interactions. A cost function is designed, the minimization of which yields the solution to the problem. Computer simulation of the proposed algorithm reveals that it is able to predict the signs of all the existing weights accurately.

Keywords-gene regulatory network, fuzzy recurrent neural network, time series gene expression data, artificial bee colony optimization.

I. INTRODUCTION

The Artificial Bee Colony optimization (ABC) [3] is a population based algorithm for optimization of continuous functions. The ABC draws inspiration from the stochastic behaviour of foraging in bees. In case of real honey bees, the self-organization dynamics is based on four properties: positive feedback (exploration through waggle dance), negative feedback (prevention of exploitation of poor food sources), fluctuations (scouting for new food source) and multiple interactions. Recent studies have suggested that ABC is better suited in terms of convergence speed to solve optimization problems than other evolutionary algorithms (EA) including the well-known differential evolution (DE) [4].We here apply the ABC algorithm to the Gene Regulatory Network (GRN) identification problem. ABC incorporates a flexible and well-balanced mechanism to adapt to the global and local exploration and exploitation abilities. Hence, this method is efficient in handling large and complex search spaces. ABC with its ability to handle combinatorial explosion appears to be very promising to the optimization problem addressed here. It can be seen from the results that, ABC based optimization model has performed quite satisfactorily in comparison with DE.

This regulatory mechanism of gene dynamics in GRN identification problem provides insight on the interaction between different genes. Currently, with the advancement of the DNA micro array technology, it has become possible to simulate gene regulatory network from gene expression time series data. Researchers are taking keen interest to model gene regulatory networks by soft computing technique, and attempted several approaches, including Boolean networks [7], Linear differential model , Bayesian networks [6], linear additive regulation models ,and the like. None of the above technique could provide a full-proof solution to the problem of gene regulatory network.

In GRN, the weight between neuron gives the numeric interaction values between genes. So, if it is possible to find those weight values between neuron from the time series data available for genes, then the real interaction between genes will be revealed. Further different techniques are available to find these interactions between neurons. For example, back propagation through time neural net [9] has been used for weight determination. Here, we have used artificial bee colony optimization algorithm to determine weights [5].

In this paper, we use the basic GRN framework as used in [5]. The present work differs from [5] with respect to the optimization procedure. Here we use ABC instead of DE as used in [5]. The performance of the current work is significantly better than the one reported in [5] considering root mean square error and convergence speed of the procedure.

ABC seems to be promising for this optimization problem because of the following reasons: 1) providing better solution quality to find out fuzzy membership distribution of weight connection between genes in GRN, 2) retrieving correct signs of almost all the weights for known networks, 3) combining local search methods with global search methods attempting to balance exploration and exploitation processes giving high speed of convergence, 4) preventing the search technique

978-1-4577-0792-6/11/$26.00 ©2011 IEEE

from premature convergence problem providing global search ability with the help of scout unit.

The paper is organized as follows. In section II, we propose the framework used in our GRN identification problem. In section III and IV, we describe the artificial bee colony optimization algorithm used to find the simulated network parameters and explain the fuzzy technique to represent the connection between genes. In section VII, we present the simulated results and in section VIII, we demonstrate the use of our model to simulate a gene regulatory network using real gene expression time series data.

II. Framework Used In Our GRN Identification Problem

A. Model used in the framework

A.1. Dynamics:

To the dynamic aspect of recurrent neural network, we have chosen the differential equation (1) as our model, where each gene expression is differentiated with respect to time.

$$T_i \frac{dg_i}{dt} = f\left(\sum_{j=1}^{N} w_{ji}^{*}\ g_j(t) - b_i\right) - k_i g_i \quad (1)$$

Let g_i is the expression of i^{th} gene, T_i is the time constant, w_{ji}^{*} be a defuzzified weight from neuron j to neuron i in the neural net representation of a recurrent neural network ; w_{ji}^{*} can be positive, negative or zero depending on whether j^{th} gene is activating, inhibiting gene i or doesn't have any effect on it at all, 'b_i', 'k_i', and 'N' represent the bias term for i^{th} gene, decay constant for g_i, and total number of genes present in the network, $f(x)=1/(1+e^{-x})$ is the nonlinear function used to get the output of each gene. Here, x is the combined effect of all genes on i^{th} gene.

To incorporate the discrete feature in our model with respect to time we have changed (1) as follows:

$$T_i \frac{dg_i}{dt} = f\left(\sum_{j=1}^{N} w_{ji}^{*}\ g_j(t) - b_i\right) - k_i g_i$$

$$T_i \frac{g_i(t+\Delta t) - g_i(t)}{\Delta t} = f\left(\sum_{j=1}^{N} w_{ji}^{*}\ g_j(t) - b_i\right) - k_i g_i(t)$$

$$g_i(t+\Delta t) = \frac{\Delta t}{T_i} f\left(\sum_{j=1}^{N} w_{ji}^{*}\ g_j(t) - b_i\right) + g_i(t)\left(1 - \frac{\Delta t}{T_i} k_i\right) \quad (2)$$

Equation (2) demonstrates how expression of a particular gene changes with time in response to the other genes present in the network.

A.2. Fuzziness of the parameters:

Here the weight w_{ji} from neuron j to i has a fuzzy membership distribution $\mu(w_{ji}^{*})$, and the corresponding fuzzy set is given by a doublet $\{w_{ji}^{k} \mid \mu(w_{ji}^{k})\}$. The w_{ji}^{*} of equation (1) is evaluated by centroidal defuzzification procedure given by (3).

$$w_{ji}^{*}(t) = \frac{\sum_{k=1}^{F} w_{ji}^{k} \times \mu\left(w_{ji}^{k}(t)\right)}{\sum_{k=1}^{F} \mu\left(w_{ji}^{k}(t)\right)} \quad (3)$$

The bias b_i, and time constant T_i are also represented in a similar manner like w_{ij}^{*} (Fig.2.) with F as the number of elements in fuzzy set.

B. Synthesis of gene expression data

Here we attempted to generate artificial gene expression time series data to test the accuracy of our method. Using the model in equation (2), we have generated time series data using the parameter values of TABLE I of a 4-gene network.

TABLE-I

PARAMETER VALUES OF A 4-GENE NETWORK						
	Gene1	Gene2	Gene3	Gene4	b_i	T_i
Gene1	20.0	-20.0	0.0	0.0	0.0	10.0
Gene2	15.0	-10.0	0.0	0.0	-5.0	5.0
Gene3	0.0	-8.0	12.0	0.0	0.0	5.0
Gene4	0.0	0.0	8.0	-12.0	0.0	5.0

The interpretation of the above weight values are as follows. From TABLE I we can see that box (1, 2) = -20.00, the meaning is that gene1 has -20.00 unit of effect on gene2. We have chosen the same set of values as in paper [1] to compare the accuracy of our model. The generated time series data for the genes are shown in Fig.1.

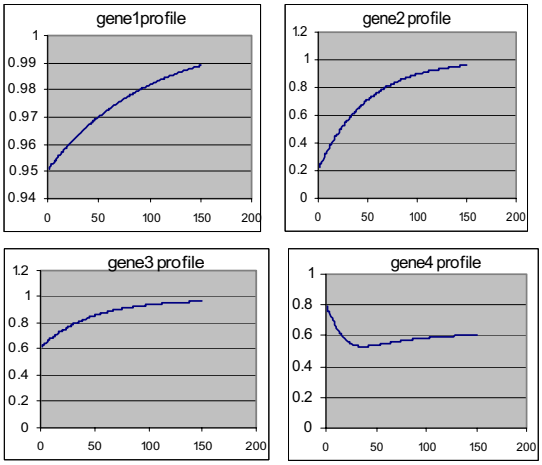

Fig.1.Expression profile of gene1, gene2, gene3, and gene4, respectively

It can be seen from Fig. 1 that nearly after150 points, the expression of all the genes get saturated, therefore this is the region from where we can extract maximum information. Because of this reason we have used 150 data points for each gene profile.

C. The cost function used

Accuracy of gene regulatory network (GRN) design mainly depends on two issues (i) how well we can measure the accuracy of the existing connection values of the network, and (ii) how well we can measure the accuracy of the skeletal structure (network topology) of the simulated network. Handling both issues simultaneously is a tough job, because we do not have

any knowledge except the available gene expression time series data, which is also limited.

To meet the first issue, we evaluated the accuracy of the produced gene expression of our simulated network by comparing it with the gene expression produced using the network parameters of TABLE I with the hope that if the network parameters of our simulated network is closer to the parameters of TABLE I then the difference (error) between these two set of gene expression will be less. That error has been calculated using the equation (4).

$$C_1 = \frac{1}{\Gamma NM}\sum_{k=1}^{M}\sum_{t=1}^{\Gamma}\sum_{i=1}^{N}\{[g_{org}^{i}(t)]_k - [g_{cal}^{i}(t)]_k\}^2 \quad (4)$$

Here M is the number of time series used; r is the number of data point in each time series data, and N is the number of gene present in the network. $[g^{i}_{org}(t)]_k$ is the original expression of i^{th} gene at t^{th} time instance in k^{th} time series, and $[g^{i}_{cal}(t)]_k$ is the calculated expression of the same using our simulated network.

The study of genetics reveals that in a gene regulatory network it is unlikely that all the genes interact with each other; rather few genes are involve in regulation of a gene. Considering this practical phenomenon, we designed the cost function given by equation (5).

$$C_2 = p\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{|w_{ij}^{*}|}{|1+w_{ij}^{*}|} \quad (5)$$

Here w_{ji}^{*} is the connection value between gene j, and gene i, and p is a constant. Choosing proper value of p is also tricky. An appropriate value will lead to a good solution or it may mislead the system, its value should be such that C_2 can't override C_1.

Our final cost function is shown in equation (6).

$$C = C_1 + C_2$$
$$C = \frac{1}{\Gamma NM}\sum_{k=1}^{M}\sum_{t=1}^{\Gamma}\sum_{i=1}^{N}\{[g_{org}^{i}(t)]_k - [g_{cal}^{i}(t)]_k\}^2 + p\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{|w_{ij}^{*}|}{|1+w_{ij}^{*}|} \quad (6)$$

Using this cost function, we will select the solution with the smallest cost value as the final solution i.e. if cost of $solution_1$ is less than that of $solution_2$ then $solution_1$ is our final solution.

III. Artificial Bee Colony Optimization Algorithm (ABC)

In ABC algorithm, the colony of artificial bees contains three groups of bees:

- A bee waiting on a dance area for making decision to choose a food source is called an onlooker.
- A bee going to the food source visited by itself previously is named as employed bee
- A bee carrying out random search is called a scout.

In ABC algorithm, the position of a food source represents a possible solution of the optimization problem and the nectar amount of a food source corresponds to the fitness of the associated solution. The number of employed bees and onlooker bees is equal to the number of solutions in the population.

ABC consists of following steps:

A. Initialization

ABC generates a randomly distributed initial population P (g=0) of N_p solutions (food source positions) where N_p denotes the size of population. Each solution X_i (i=0, 1, 2,…, N_p-1) is a *D* dimensional vector. Here *D* is the number of optimization parameters.

B. Placement of employed bees on the food sources in the memory

An employed bee produces a modification on the position (solution) in her memory depending on the local information (visual information) as stated by equation (8) and tests the nectar amount (fitness value) of the new source (new solution). Provided that the nectar amount of the new one is higher than that of the previous one, the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one in her memory.

C. Placement of onlooker bees on the food sources in the memory

After all employed bees complete the search process; they share the nectar information of the food sources (solutions) and their position information with the onlooker bees on the dance area. An onlooker bee evaluates the nectar information from all employed bees and chooses a food source depending on the probability value associated with that food source, p_i, calculated by the following expression:

$$p_i = \frac{fit_i}{\sum_{i=0}^{Np-1} fit_i} \quad (7)$$

where fit_i is the fitness value of the solution *i* evaluated by its employed bee, which is proportional to the nectar amount of the food source in the position *i* and N_p is the number of food sources which is equal to the number of employed bees. After that, as in case of employed bee, onlooker bee produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution). Providing that its fitness is better than that of the previous one, onlooker bee memorizes the new position and forgets the old one.

Neighborhood food source has been generated by altering the value of one randomly chosen

solution parameter and keeping other parameters unchanged. Let us notate the solution X_i and let us suppose that the solution X_i has D parameters with values $x_{i0}, x_{i1}, x_{i2}, \dots, x_{i(D-1)}$. In order to find a solution X_i' in the neighborhood of X_i, a solution parameter j and another solution X_k are selected on random basis. Except for the value of chosen parameter j, all other parameter values of X_i' are same as in the solution x_i, for example,

$X_i' = (x_{i0}, x_{i1}, x_{i2}, \dots, x_{i(j-1)}, x_{ij}', x_{i(j+1)}, \dots, x_{i(D-1)})$

The value of x_{ij}' parameter in X_i' solution is computed using the following expression:

$$x_{ij}' = x_{ij} + u(x_{ij} - x_{kj}) \quad (8)$$

where u is a uniform variable in [-1, 1] and k is any number between 0 to N_p but not equal to i.

If a parameter produced by this operation exceeds its predetermined limit, the parameter can be set to an acceptable value.

D. Send scouts to search area for discovering the new food sources in the memory. In the ABC algorithm, if a position cannot be improved further through a predefined number of cycles called 'limit', the food source is abandoned. This abandoned food source is replaced by the scouts by randomly producing a position.

After that again steps (B), (C) and (D) will be repeated until the stopping criteria is met.

VI. Extraction Of Fuzzy Membership Distribution Of Weights Using ABC

In our paper, we have used the well known artificial bee colony (ABC) optimization algorithm to find the simulated network. To spread the initial candidate solutions as far possible in the search space with the hope that some of the solutions may be close to the original solution we have used a chaos system [2] in ABC. The process of producing the chaos is as follows:

$$Z_{k+1} = \mu Z_k (1 - Z_k) \quad (9)$$

where k = 0, 1, 2, 3… Θ, Θ is the number of chaotic iteration, μ is the control parameter. Z_k takes any value between 0 and 1; it is the selected value in the k^{th} iteration. When we set μ=4 and $Z_0 \neq \{0, 0.25, 0.5, 0.75, 1\}$ then the value of Z_k distributes with proper randomness and irregularity. We indeed found that this initialization improve the overall convergence rate of the artificial bee colony optimization algorithm.

Each individual food source of artificial bee colony optimization algorithm represents a complete solution. As an example, one food source of the above 4-gene network contains 16 connection weights, 4 bias terms, and 4 time constants with total of 24 fields. Each field of every individual represented by a fuzzy set, as number of elements= C_f. The population pool of the artificial bee colony optimization algorithm for the 4-gene network with C_f=5 can be represented pictorially as in Fig. 2.

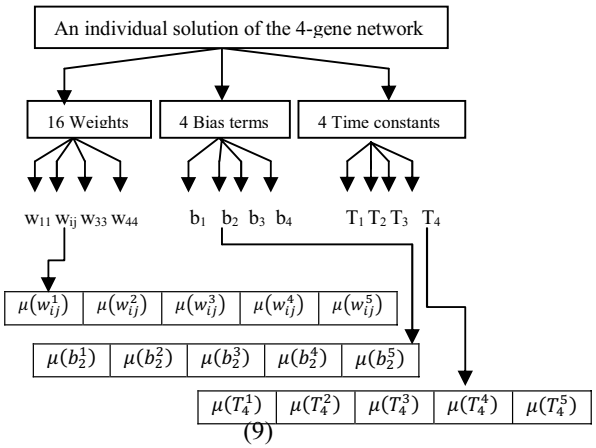

Fig. 2. Pictorial representation of a single individual food source of ABC

In Fig. 2. C_f=5, $\mu(w_{ij}^1)$, , $\mu(w_{ij}^2)$, , $\mu(w_{ij}^3)$, , $\mu(w_{ij}^4)$, $\mu(w_{ij}^5)$ represents the fuzzy membership values of the weights , w_{ij} of any individual food source. $\mu(b_2^1)$, $\mu(b_2^2)$, $\mu(b_2^3)$, $\mu(b_2^4)$, $\mu(b_2^5)$ represent the fuzzy membership values of the bias terms b_2, and $\mu(T_4^1)$, $\mu(T_4^2)$, $\mu(T_4^3)$, $\mu(T_4^4)$, $\mu(T_4^5)$ represent the fuzzy membership of the time constants T_4.

VII. Results

The results are shown in TABLE- II and III. The results are generated using 4 time series data. We have chosen 1000 iterations of the artificial bee colony optimization algorithm, 300 iterations for chaotic initialization algorithm. Our algorithm takes nearly 57 minutes on a Pentium dual port computer containing 2 GB RAM.

TABLE-II

RESULTS AFTER RUN-1 USING 4 TIME SERIES DATA, 50 POPULATION C_f=7 USING ABC

	Gene1	Gene2	Gene3	Gene4	bi	Ti
Gene1	17.39	-22.54	0.00	0.36	9.00	9.8
Gene2	4.61	-4.83	0.55	1.82	-14.69	4.8
Gene3	1.33	-1.42	21.62	0.00	-6.19	5.0
Gene4	0.00	1.02	0.00	-12.96	9.29	8.9

TABLE-III

RESULTS AFTER RUN-2 USING 4 TIME SERIES DATA, 70 POPULATION C_f=5 USING ABC

	Gene1	Gene2	Gene3	Gene4	bi	Ti
Gene1	20.00	-19.92	0.68	0.00	-17.86	10.7
Gene2	15.74	-17.12	0.70	4.39	-12.56	5.02
Gene3	2.42	7.44	12.18	5.80	0.00	5.14
Gene4	0.00	0.00	14.22	-28.18	6.19	6.04

As can be seen from above tables the connection weights are identified sufficiently accurately, all the negative and positive signs and some nonexistent weights are identified correctly in TABLE-II, though signs of some of the weights are not correct in TABLE-III, at this point we want to remind the reader once again that we don't had any prior knowledge about the network structure. The results can be shown as below for better understanding:

TABLE-IV

SIGNS OF TABLE-I

	Gene1	Gene2	Gene3	Gene4	bi	Ti
Gene1	+	-	0.0	0.0	0.0	10.0
Gene2	+	-	0.0	0.0	-	5.0
Gene3	0.0	-	+	0.0	0.0	5.0
Gene4	0.0	0.0	+	-	0.0	5.0

TABLE-V

SIGNS OF TABLE-II

	Gene1	Gene2	Gene3	Gene4	bi	Ti
Gene1	+	-	0.00	0.36	9.00	9.82
Gene2	+	-	0.55	1.82	-	4.82
Gene3	1.33	-	+	0.00	-6.19	5.09
Gene4	0.00	1.02	+	-	9.29	8.92

TABLE-VI

SIGNS OF TABLE-III

	Gene1	Gene2	Gene3	Gene4	bi	Ti
Gene1	+	-	0.68	0.00	-17.8	10.7
Gene2	+	-	0.70	4.39	-	5.02
Gene3	2.42	+(wrong)	+	5.80	0.00	5.14
Gene4	0.00	0.00	+	-	6.19	6.04

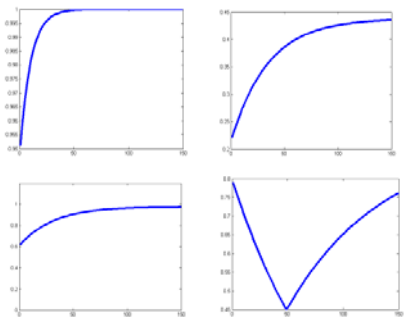

Fig. 3. Expression profile of gene1, gene2, gene3, and gene4 obtained using ABC using parameters of TABLE-II

The relative performance of ABC and DE- based simulation for inferring about the parameters of gene regulatory network can be studied through the parameters obtained in DE-based simulation (TABLE-VII) and the corresponding signs of the obtained weights (TABLE-VIII).The gene expression profile obtained using the parameters of TABLE-II are given in Fig. 3.

TABLE-VII

RESULTS AFTER RUN-1 USING 4 TIME SERIES DATA, 50 POPULATION C_f=7 USING DE

	Gene1	Gene2	Gene3	Gene4	bi	Ti
Gene1	14.48	-15.43	2.57	1.14	2.59	16.3
Gene2	9.21	-17.06	8.90	15.68	-3.76	4.56
Gene3	12.05	-8.54	8.52	5.37	-3.76	3.03
Gene4	6.19	0.00	2.56	-10.78	-7.20	22.4

TABLE-VIII

SIGNS OF TABLE-VII

	Gene1	Gene2	Gene3	Gene4	bi	Ti
Gene1	+	-	2.57	1.14	2.59	16.3
Gene2	+	-	8.90	15.68	-	4.56
Gene3	12.05	-	+	5.37	-3.76	3.03
Gene4	6.19	0.00	+	-	-7.20	22.4

The performance metric used here to determine how close the estimated parameters are close to the original values of connection weights, bias terms, and time constants is Root Mean Square Error (RMSE) given as

$$RMSE = \sqrt{\frac{1}{n}(org_i - cal_i)^2} \quad (9)$$

Where, n=number of parameters= 24(here), org_i and cal_i are the original and calculated values of the i-th parameter, i=1, 2, ... 24.

Using TABLE-IV, V and VIII, it is apparent that for a population of size 40 and C_f=7

$$RMSE_{ABC} = 0.6692 \quad (10)$$

$$RMSE_{DE} = 0.9602 \quad (11)$$

Comparing TABLE-V and VIII, along with equation (10) and (11) it is apparent that ABC has outperformed DE in inferring actual signs of the weights in the simulated network.

In Fig. 4 and 5 we have plotted the cost function value of the best solution obtained in each step of ABC and DE-based simulation. It is noted that in both the cases ABC converges faster than DE. Also, for a fixed number of iteration ABC provides better solution than DE.

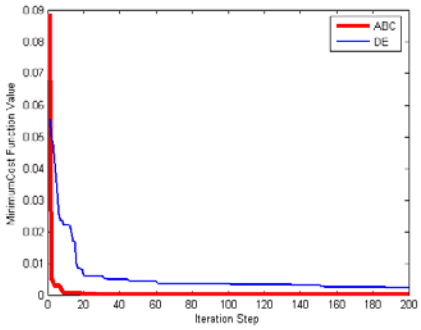

Fig. 4. Minimum cost function value in each step in ABC and DE in case of 200 iteration

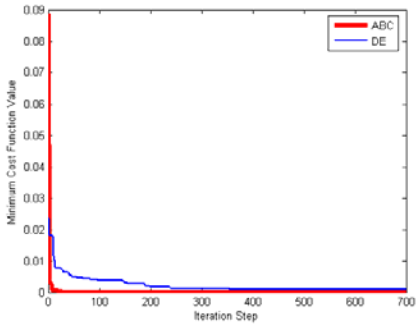

Fig. 5. Minimum cost function value in each step in ABC and DE in case of 700 iteration

VIII. INFERRING GRN USING REAL DATA SET

We have used our model to infer the gene regulatory network of e.coli. Bacteria S.O.S DNA repair network, this network consists of nearly 30 genes regulated at the transcription level. Four experiments have been conducted with different UV light intensities and eight major genes have been documented, these genes are uvrD, lexA, umuD, recA, uvrA, uvrY, ruvA,

polB. This regulatory mechanism can be shown pictorially as in Fig. 6.

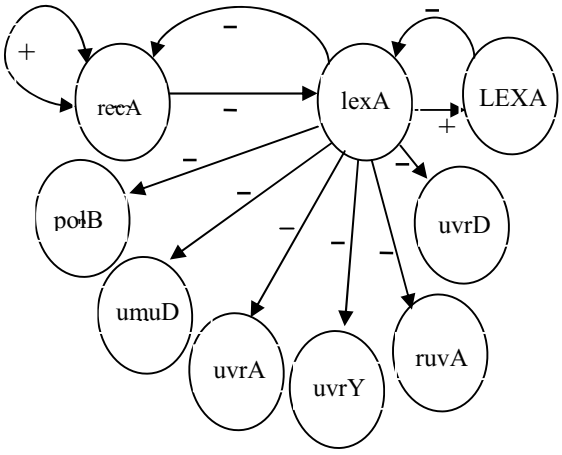

Fig. 6. E.coli S.O.S. DNA repair network, activation is represented by '+' sign and inhabitation by'-', genes are written with small letter and protein with capital letter

Gene profiles of E.coli S.O.S DNA repair network consists of 50 data points, sampled at every 6 minutes. The data set consists of four 8 x 50 matrix, each column represents the observation of expression value at a particular time instant of eight genes , each row represents the fifty expression value of a particular gene at different time instants. This data set is available in the website [http://www.weizmann.ac.il/mcb/UriAlon/]. It is one of the best data set which fit with our model. We have conducted same experiment as with the above artificial data, the identified interaction values between genes are represented in the following table:

TABLE-IX

IDENTIFIED INTERACTION VALUES OF E.COLI S.O.S. DNA REPAIR NETWORK								
	uvrD	lexA	umuD	recA	uvrA	uvrY	ruvA	polB
uvrD	-14.0	-11.2	-19.0	-2.6	-6.9	-14.7	4.4	-28.1
lexA	-10.6	-15.1	-21.7	25.5	5.7	-15.8	-3.3	-1.8
umuD	-16.2	,-2.1	-9.9	-21.0	10.9	6.4	-3.3	-2.9
recA	0.00	-14.4	-19.8	2.5	-24.5	-5.3	-11.9	3.9
uvrA	3.25	-11.3	0.0	-10.9	6.4	0.0	19.7	0.0
uvrY	17.2	3.0	-11.1	0.0	0.0	0.0	-2.2	0.0
ruvA	-2.10	-26.7	-11.3	13.2	0.0	-15.1	-4.6	0.0
polB	-8.02	12.4	15.6	13.3	23.5	0.0	-21.6	12.9

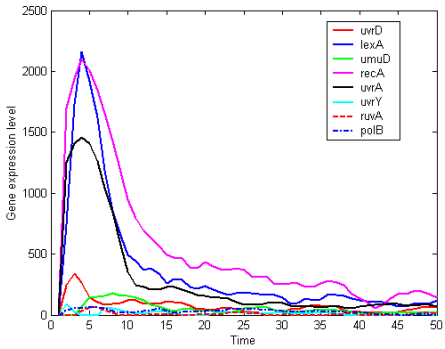

Fig. 7. The measured gene expression profile using parameters of TABLE-IX

We have fixed the lower and upper range of the interaction value as -30 and +30; the values of other parameters of the algorithm are as used in our model. The result of our algorithm has been shown in TABLE-IX. Here the reader should keep in mind that the cost function is used contains multi minima and also the available gene expression time series data contains only 50 data points.

CONCLUSION

Since ABC algorithm uses exploitative process efficiently to converge minima and explorative process to provide sufficient diversity in the population for a given colony size, ABC algorithm does not need big number of colony size to solve optimization problems with high dimensions. Control parameter *limit* is the core parameter of the algorithm dictating the occurrence of scout bees that are responsible for providing the diversity in the population. Also, ABC algorithm is not very sensitive to initialize parametric ranges compared to other algorithm considered in this paper.

Due to the large number of model parameters and the small number of data sets available, the system of equations in GRN identification problem is highly under-determined and ambiguous. Therefore, multiple solutions exist, which fit the given data, but show only little resemblance with the original target system.

REFERENCE

[1]: Rui Xu, Donald C. Wunsch II and Ronald L. Frank, " Inference of genetic regulatory networks with recurrent neural network models using particle swarm optimization" IEEE/ACM transaction on computational biology and bioinformatics,vol.4, No.4,pp 681-692, 2007.

[2]: C. Lng, S.Q. Li. Chaotic spreading sequences with multiple access performance better than random sequences. IEEE transaction on Circuit and System -I, Fundamental Theory and Application, 47(3):394-397, 2000.

[3]: B. Basturk, Dervis Karaboga, "An Artificial Bee Colony (ABC) Algorithm for Numeric function Optimization", IEEE Swarm Intelligence Symposium 2006, May 12-14, 2006, Indianapolis, Indiana, USA.

[4]: D. Karaboga, B. Basturk, "On the performance of artificial bee colony (ABC) algorithm", In: Applied Soft Computing 8(2008) 687-697.

[5]: D. Datta, A. Konar, R. Janarthanan, "Extraction of interaction information among genes from gene expression time series data", Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress, 9-11 Dec, 2009, 98-103.

[6]: B. Perrin, L. Ralaivola, A. Mazurie, S. Battani, J. Mallet, and F.d'Alche´-Buc, "Gene Networks Inference Using Dynamic Bayesian Networks," Bioinformatics, vol. 19, pp. ii138-ii148, supplement 2, 2003.

[7]: T. Akutsu, S. Miyano, S. Kuhara, "Identification of genetic networks from a small number of gene expression patterns under the Boolean network model," Pac Symp Biocomput, 17-28, 1999.

[8]: S. Liang, S. Fuhrman, R. Somogyi, "Reveal, a general reverse engineering algorithm for inference of genetic network architectures," Pac Symp Biocomput, 18-29, 1998.

[9]: P. Werbos, (1995) "Backpropagation through time what it does and how to do it," Proceedings of IEEE, 78(10), 1550-1986.