Genetic Regulatory Network Inference using Recurrent Neural Networks trained by a Multi Agent System

Adel Ghazikhani
Computer Engineering Department
Ph.D. student, Ferdowsi University of Mashhad and Lecturer, Engineering department Imam Reza University
Mashhad a_ghazikhani@ieee.org

Mohammad Reza Akbarzadeh T
Center for Applied research on Intelligent Systems and Soft Computing, Ferdowsi University of Mashhad, Iran
akbarzadeh@ieee.org

Reza Monsefi
Computer engineering department
Ferdowsi University of Mashhad, Iran
monsefi@um.ac.ir

Abstract — **We propose a novel algorithm for gene regulatory network inference. Gene Regulatory Network (GRN) inference is approximating the combined effect of different genes in a specific genome data. GRNs are nonlinear, dynamic and noisy. Time-series data has been frequently used for GRN modeling. Due to the function approximation and feedback nature of GRN, a Recurrent Neural Network (RNN) model is used. RNN training is a complicated task. We propose a multi agent system for RNN training. The agents of the proposed multi agent system trainer are separate swarms of particles building up a multi population Particle Swarm Optimization (PSO) algorithm. We compare the proposed algorithm with a similar algorithm that uses RNN with standard PSO for training. The results show improvements using the E. coli SOS dataset.**

Keywords- Gene Regulatory Network Inference, Particle Swarm Optimization, Multi Population PSO, Recurrent Neural Networks, Multi Agent Systems

I. INTRODUCTION

The genomic revolution is seeking its path to understand the complex genetic relations inside living organisms. One of the substantial obstacles of this path is the interaction among genes and proteins as the two main components present in any living organism. The first step in understanding the interaction among genes and proteins is determining the interaction among each component individually. In this research we are focusing on the relations between genes known as Genetic Regulatory Network (GRN).

GRNs could be inferred by different approaches. One scheme is experimental techniques. Experimental techniques are usually difficult and limited. One other approach for GRN inference is incorporating mathematical models. In this research we are concentrating on computational approaches which use mathematical models to infer GRNs.

Computational approaches for GRN inference rely on data gathered from GRN experiments. The main form of data for GRN inference is Microarrays [1].

Much research has been done on GRN learning. One of the first methods used for GRN learning was neural networks. Mjolsness et. al. [2,3] used differential equations to model GRNs and used Recurrent Neural Networks (RNN) to learn the GRN. In this method gene regulation was represented as a combination of (i) cis-acting regulation by the promoter, and (ii) trans-acting regulation by the TF products of other genes. Later on, Vohradsky used RNN [4] but he assumed that GRN's have Multi-genic regulation, including positive and/or negative feedback.

Another paradigm used to model GRN is Fuzzy sets. Woolf & Wang [5], used Fuzzy rules to transform the gene expression values into qualitative descriptors. Sokhansanj et. al. [6], used a scalable linear variant of fuzzy logic to learn GRN. Du et al [7] used multi scale fuzzy c-means clustering. In this method domain knowledge from different sources was used.

Besides this Evolutionary computation was used for GRN learning too. S. Kikuchi et al. [8] used GA to learn the best GRN that fits the data. Qian et al. [9] modeled GRN with a nonlinear differential equations and used GP with kalman filter to learn this model. Another group of methods are hybrids which usually combine two or more of the above paradigms. Ritchie et al. [10] used multi layer perceptron with GP to learn GRN from microarray time series data. Xu et al. [11] modeled GRN using RNN and then used particle swarm optimization to train the network. In this method the training is done for structures as well as parameters.

Since GRNs have a nonlinear and feedback nature we use a RNN model. RNN training is a complicated task. We propose a Multi Agent System (MAS) for RNN training. In the proposed MAS the agents are particle swarms. We aim to enhance the results of GRN inference by training the RNN more precisely.

The paper is organized as follows. Section II discusses some preliminary paradigms used in the main algorithm. Section III describes our proposed algorithm for inferring network interactions. Section IV presents the experimental results, and Section V concludes the paper.

II. BACKGROUND

A. RNN model for GRN

The model we have used for our genetic regulatory network is an RNN similar to [11] (Fig. 1).

$$e_i(t+\Delta t) = \frac{\Delta t}{t} \times f\left(\sum_{j=1}^{N} w_{ij} e_j(t) + \beta_i\right) + \left(1 - \frac{\Delta t}{\tau_i}\right) e_i(t) \quad (1)$$

In the aforementioned model f is a nonlinear function (usually sigmoid; $f(z) = 1/(1+e^{\tau}(-z)))$), e is the expression level of a gene, w_{ij} is the effect of the jth gene on the ith gene. τ is the time constant and β is the bias term.

B. RNN training

RNN training is a complicated task. Different approaches such as Back Propagation Through Time (BPTT) [12] and Evolutionary Algorithms (EAs) [13] have been used. Here we propose to use multi population particle swarm optimization to train RNN.

The training has two parts i.e. structure and parameter training. By structure training we mean the optimized structure of the RNN and by parameter training we mean the optimized parameters of the model [11].

C. Multi population PSO

PSO is an evolutionary algorithm for optimization of complex objective function [14]. It is inspired from the behavior of swarms of birds seeking for food. The main feature of this algorithm is that it incorporates local information as well as global information.

One of the problems of PSO is premature convergence. This occurs when the results are suboptimal. One of the methods used for preventing premature convergence is multi population PSO [15].

In multi population PSO we divide the whole population into some sub populations that evolve independently. Each sub population exchanges evolution information with other sub populations after some generations. This method can avoid the premature convergence problem of PSO effectively.

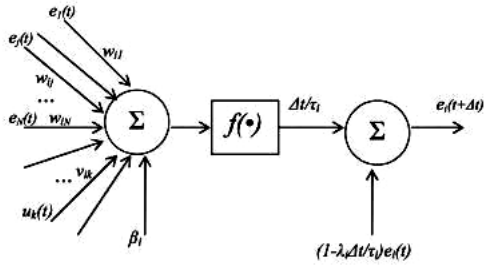

Figure 1. A neuron in the RNN mode [11]

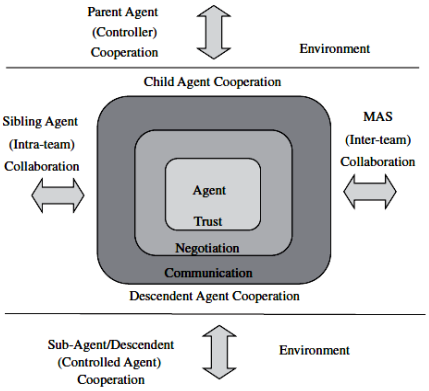

Figure 2. The Trust, Negotiation and Communication(TNC) model for MAS[19]

D. Multi agent systems

A Multi Agent System (MAS) is a system that aims to solve problems in a distributive manner [16]. The distributedness is achieved through the organization, cooperation and coordination of multiple autonomous agents. MASs have different architectures such as BDI (belief, desire, and intension), Layered and others. A newer architecture is TNC.

E. TNC architecture of MAS

The Trust, Negotiation and Communication (TNC) model for MAS architecture is based on the concept of trust [17, 18]. Agents interact with each other according to the trust they have in each other. In the TNC model trust is conceptualized as a piece of information that is exchanged among the agents in the MAS. Similar to human societies in a TNC based MAS, trust is a dynamic concept i.e. it could increase or decrease depending on the situation the agent is experiencing. The trust between two agents increases when they have successful interactions and decreases when they don't. Therefore the main issue of the TNC model is how to define trust in MAS. This model is shown in Fig. 2[19].

III. THE PROPOSED ALGORITHM

The proposed algorithm is a novel method for gene regulatory network inference. The main contribution is the usage of MAS to train a RNN for GRN. The whole algorithm is shown in Fig. 3.

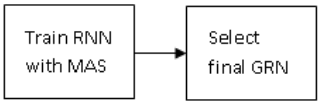

Figure 3. The whole algorithm

As shown in Fig. 3 the whole algorithm has two main elements i.e. the RNN trainer and the final network selection. In the upcoming sections we clarify these elements with detail.

A. MAS for RNN training

In this paper we use MAS to train the RNN which models the GRN. As mentioned in section 2E, F every MAS has an architecture and a scenario for cooperation between agents. These issues are explained next.

1) MAS architecture

With respect to the TNC model the proposed MAS consists of three types of agents. The agents are the parent agent, the structure PSO agent and the parameter PSO agent (Fig. 4). The parent agent starts the training algorithm and finally outputs the results. The structure PSO agent commits the RNN structure training. Lastly the parameter PSO agent evolves the parameters of the RNN model i.e. RNN parameter training. The MAS could be thought of as a multi population PSO algorithm. Next we will discuss the MAS scenario, i.e. what happens in the MAS.

2) MAS scenario

The MAS is initialized by the parent agent. This agent starts two structure PSO agents. Every structure PSO agent starts three parameter PSO agents in each iteration. In this process the global best fitness of the parameter PSO agents is sent to the structure PSO agents as the ***trust*** value. The structure PSO agent raises a first-price sealed bid auction and chooses the maximum trust i.e. the best particle. The structure PSO agent sets the fitness of its particles using the fitness of this particle. Later on, again the structure PSO agents send their global best fitness to the parent agent as the ***trust*** value. The parent agent raises a first-price sealed bid auction and chooses the maximum trust. Therefore in each run the best network is determined in a multi agent paradigm using the concept of trust.

As we mentioned earlier trust in each of the interfaces between the agents is defined as the fitness of the global best particle.

$$Trust_i = Global\ fitness\ particle_i \quad (2)$$

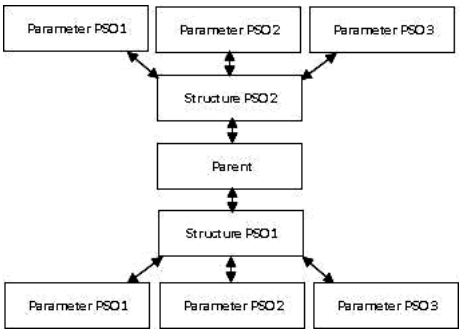

Figure 4. The multi agent system aimed to train the RNN

B. Final network selection

Due to the sparseness and limitation of microarray time series data, it seems inadequate to determine a specific network with high fitness value as the global best network. Therefore a probabilistic approach is used to determine the final network. We have used a method similar to [11] to determine the final network. This probabilistic approach has two steps. First we define the number of connections and then define the connections themselves. The result of the algorithm is M networks. To infer the final network there is two steps. Firstly, a heuristic based on the assumption that the number of times that a nonexistent connection is observed m in the inferred networks follows a binomial distribution with a probability p is used [23]. The probability is set less than or equal to 0.05.

$$P(m) = C_M^m p^m (1-p)^{M-m} 0.5 \quad (3)$$

In Eq. (3), p is set between 0 and 1, deducing the m that satisfies Eq. (3). After that we count the number of connections that occur more than m times as NC. Therefore we can draw a plot reflecting the relation of p and NC. Using this plot we determine the number of connections as the NC that corresponds to an abrupt change (NC*). The second step is to determine the connections where we choose the first NC* connections in the list of the most frequent connections.

IV. EXPERIMENTAL RESULTS

A. Implementation

Since MAS have parallel functionality, they have a complicated implementation. Here we use JACK software [20] to implement the MAS. We train our RNN using the MAS. After training the RNN, Matlab is used to determine the final network.

B. Experiments

We evaluate the proposed algorithm using the E. coli SOS DNA repair network dataset [21] (Fig. 5). The dataset consists of four subsets. Each of the subsets is a time series composed of 50 time points.

The evaluation is performed in two settings: 1) single time series, 2) multiple time series. We compare the proposed algorithm with Xu's algorithm [11], which is a recent similar algorithm.

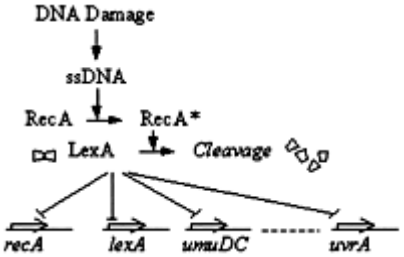

Figure 5. S.O.S. DNA Repair network. Activations are represented by arrows (→) and inhibitions by T (). Genes initials are in lower cases, proteins in capital letters[21]

To finalize the results we have used the genetic regulatory network regulations referenced in [22]. The true regulations of the genetic regulatory network are lexA gene inhibits all the other genes and gene recA activates gene lexA. Therefore there are sum of nine true regulations.

We have used 15 individuals and 10 iterations for each of the PSO agents in both algorithms. The algorithms were run 10 times and consequently the overall network was estimated using 10 networks. The results are shown in Table I.

As shown in Table I, the multi agent system has much more capability in training the RNN model. The result of Table I are different from [11]. This is because we implemented both algorithms with JACK software. In Table 2 we observe the trust values of the structure PSO agents selected after raising the first-price sealed bid auction between the structure PSO agents. As shown in Table 2 the fitness values decrease when we have multiple time series.

As we mentioned in 3B a probabilistic approach is used to specify the final networks. In Fig. 6 we observe the plots used to decide the number of connections.

TABLE I. RESULTS OF THE ALGORITHMS

Algorithm	Single time series	Multiple time series
Multi agent system	*2 true positive regulations* *lexA inhibits recA* *lexA inhibits polB*	*2 true positive regulations* *lexA inhibits recA* *lexA inhibits uvrY*
RNN-PSO [11]	*no true positive regulations*	*no true positive regulations*

TABLE II. THE BEST TRUST VALUES OF THE STRUCTURE PSO AGENTS

Runs	Single time series from SOS	Multiple time series from SOS
1	*195508.9362*	*60546.6390*
2	*195510.4513*	*60569.2943*
3	*195485.0170*	*60529.2428*
4	*195557.2553*	*60540.6206*
5	*195510.1368*	*60543.3990*
6	*195576.6669*	*60540.5526*
7	*195500.0193*	*60527.5900*
8	*195484.8248*	*60540.6474*
9	*195509.4155*	*60541.8402*
10	*195534.8426*	*60540.4808*

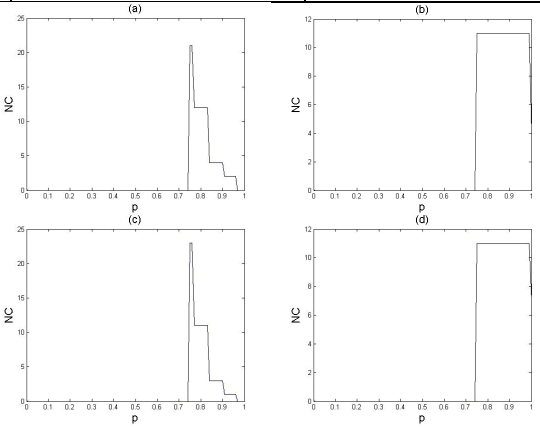

Figure 6. NC values against probabilities for four states. From top left to right, a)multi agent system with single time series, b)RNN-PSO[11] with single time series, c)multi agent system with multiple time series, d)single agent system with single time series

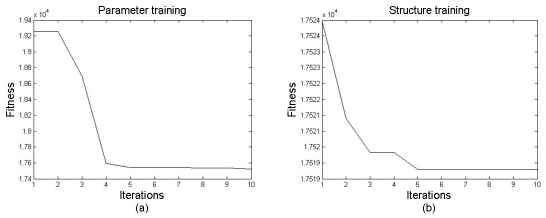

Figure 7. A view of Parameter training and Structure training. The fitness decreases as the particles evolve

For choosing the NC value that shows an abrupt change in the plot we used the largest change in the plot's slope.

In Fig. 7 we see an instance of the parameter training and structure training done in the proposed algorithm. The fitness function was an error rate so it is decreasing.

V. CONCLUSION

We proposed a novel algorithm for GRN inference. The algorithm is based on a RNN model. The main contribution is a novel method for RNN training based on a Multi Agent System (MAS). The architecture of the MAS is TNC [19]. The MAS is comprised of three types of agents, parent agent, structure PSO agent and parameter PSO agent. The structure PSO agents evolve the structures of the RNN in a multi population PSO paradigm. The parameter PSO agents do the same for the parameters of the RNN model. The proposed algorithm has better results than a similar algorithm proposed recently.

Further improvements could be made to the multi agent system to have more cooperation between the agents.

REFERENCES

[1] Special Issue on Bioinformatics, IEEE Comput., vol. 35, no. 7, Jul. 2002.

[2] E. Mjolsness, D. H. Sharp, and J. Reinitz, "A connectionist model of development," Journal of Theoretical Biology, vol. 152, pp. 429–453, 1991.

[3] E. Mjolsness, "Trainable gene regulation networks with applications to Drosophila pattern formation," in Compuational Methods for Molecular and Cellular Biology (J. M. Bower and H. Bolouri, eds.), pp. 101–117, Berlin / Heidelberg: MIT Press, 1999.

[4] J. Vohradsky, "Neural network model of gene expression," FASEB Journal, vol. 15, pp. 846–854, 2001.

[5] P. J. Woolf and Y. Wang, "A fuzzy logic approach to analyzing gene expression data," Physiol. Genomics, vol. 3, pp. 9–15, 2000.

[6] B. A. Sokhansanj, J. P. Fitch, J. N. Quong, and A. A. Quong, "Linear fuzzy gene network models obtained from microarray data by exhaustive search," BMC Bioinformatics, vol. 5, p. 108, 2004.

[7] P. Du, J. Gong, E. S. Wurtele, and J. A. Dickerson, "Modeling gene expression networks using fuzzy logic," IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, vol. 35, pp. 1351–1359, 2005

[8] S. Kikuchi, D. Tominaga, M. Arita, K. Takahashi, and M. Tomita, "Dynamic modeling of genetic networks using genetic algorithm and S-system," Bioinformatics, vol. 19, pp. 643–650, 2003

[9] L. Qian,H. Wang and E. R. Dougherty, Inference of Noisy Nonlinear Differential Equation Models for Gene Regulatory Networks Using Genetic Programming and Kalman Filtering, IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 56, NO. 7, JULY 2008

[10] M. D. Ritchie, B. C. White, J. S. Parker, L. Hahn, and J. H. Moore, "Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases," BMC Bioinformatics, vol. 4, pp. 28–36, 2003

[11] R. Xu, D. C. Wunsch, R L. Frank , Inference of Genetic Regulatory Networks with Recurrent Neural Network Models Using Particle Swarm Optimization , IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, VOL. 4, NO. 4, OCTOBER-DECEMBER 2007

[12] P.J. Werbos, "Backpropagation through Time: What It Does and How to Do It," Proc. IEEE, vol. 78, no. 10, pp. 1550-1560, 1990.

[13] D. Fogel, "An Introduction to Simulated Evolutionary Optimization," IEEE Trans. Neural Networks, vol. 5, no. 1, pp. 3-14, 1994.

[14] J. Kennedy, R. Eberhart, and Y. Shi, Swarm Intelligence. Morgan Kaufmann, 2001.

[15] L. Su-hua, W. Yao-wu, X. Xin-yin, TU Guang-yu, A Parallel PSO Approach to Multi-Objective Reactive Power Optimization with Static Voltage Stability Consideration,IEEE 2006

[16] M. Wooldridge, An Introduction to Multi Agent Systems, John Wiley & Sons, 2002

[17] J. Tweedale, P. Cutler, Trust in multi-agent systems, in: The 10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, 2006, pp. 479–485

[18] H. Kamal, J. Tweedale, U. Pierre, J. Lakhmi, Intelligent decision support system in defence maintenance methodologies, in: The Second International Conference on Emerging Technologies, 2006, pp. 560–567.

[19] A. Quteishat, C. Peng Lim, J. Tweedale and L. C. Jain, "A neural network-based multi-agent classifier system," in Neurocomputing. 2008

[20] http://www.aosgrp.com

[21] M. Ronen, R. Rosenberg, B. Shraiman, and U. Alon, "Assigning Numbers to the Arrows: Parameterizing a Gene Regulation Network by Using Accurate Expression Kinetics," Proc. Nat'l Academy of Sciences USA, vol. 99, no. 16, pp. 10555-10560, 2002

[22] 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.

[23] J. Xu and B. Nelson, personal communications, Dept. of Industrial Eng. and Management Sciences, Northwestern Univ., 2006