Hindawi Publishing Corporation
Scientifica
Volume 2016, Article ID 1060843, 14 pages
http://dx.doi.org/10.1155/2016/1060843

Research Article

Construction of Gene Regulatory Networks Using Recurrent Neural Networks and Swarm Intelligence

Abhinandan Khan,[1] Sudip Mandal,[1] Rajat Kumar Pal,[1] and Goutam Saha[2]

[1]*Department of Computer Science and Engineering, University of Calcutta, Acharya Prafulla Chandra Roy Siksha Prangan, JD-2, Sector III, Salt Lake City, Kolkata, West Bengal 700 098, India*
[2]*Department of Information Technology, North Eastern Hill University, Umshing-Mawkynroh, Shillong, Meghalaya 793 022, India*

Correspondence should be addressed to Abhinandan Khan; khan.abhinandan@gmail.com

Received 28 December 2015; Revised 19 April 2016; Accepted 24 April 2016

Academic Editor: Matthias Futschik

We have proposed a methodology for the reverse engineering of biologically plausible gene regulatory networks from temporal genetic expression data. We have used established information and the fundamental mathematical theory for this purpose. We have employed the Recurrent Neural Network formalism to extract the underlying dynamics present in the time series expression data accurately. We have introduced a new hybrid swarm intelligence framework for the accurate training of the model parameters. The proposed methodology has been first applied to a small artificial network, and the results obtained suggest that it can produce the best results available in the contemporary literature, to the best of our knowledge. Subsequently, we have implemented our proposed framework on experimental (*in vivo*) datasets. Finally, we have investigated two medium sized genetic networks (*in silico*) extracted from GeneNetWeaver, to understand how the proposed algorithm scales up with network size. Additionally, we have implemented our proposed algorithm with half the number of time points. The results indicate that a reduction of 50% in the number of time points does not have an effect on the accuracy of the proposed methodology significantly, with a maximum of just over 15% deterioration in the worst case.

1. Introduction

With the ongoing evolution of technology, massive amounts of temporal genetic expression data for different diseases are becoming available to researchers. The analysis of these data can potentially reveal many unknown cellular activities of living organisms [1, 2]. These data have enough hidden information embedded in them that if suitably analysed can revolutionise biological science and its allied applications like drug design. Accordingly, this has attracted and motivated the research fraternity to undertake detailed investigations in this domain and subsequently develop computational tools required for biologically credible analysis of these data [3–6]. In this paper, we have examined the reverse engineering of gene regulatory networks (GRNs) from temporal genetic expression datasets. These types of datasets contain crucial underlying information concerning the network dynamics among the genes (through protein).

A GRN represents the complex interregulatory relationships among genes. The transcriptional regulation of genes by other genes involves DNA, RNA, and protein as well as other molecules. The genetic interactions are indirect; that is, a gene does not interact with other genes directly. The indirect interactions take place with the help of proteins (a.k.a. transcription factors). The regulatory relationships (depending on the nature of the control) may be of two types, namely, *activation* (where there is an increase in the expression value of the target gene) and *repression* (where the expression value of the target gene decreases). The various processes involved in genetic regulation have been shown in Figure 1.

Genetic expression datasets deal with the expression values of a vast number of interacting genes. Moreover, the number of genes in a dataset is generally two to three times more than the number of time points, at the very least. This imposes a well-known computational problem known as

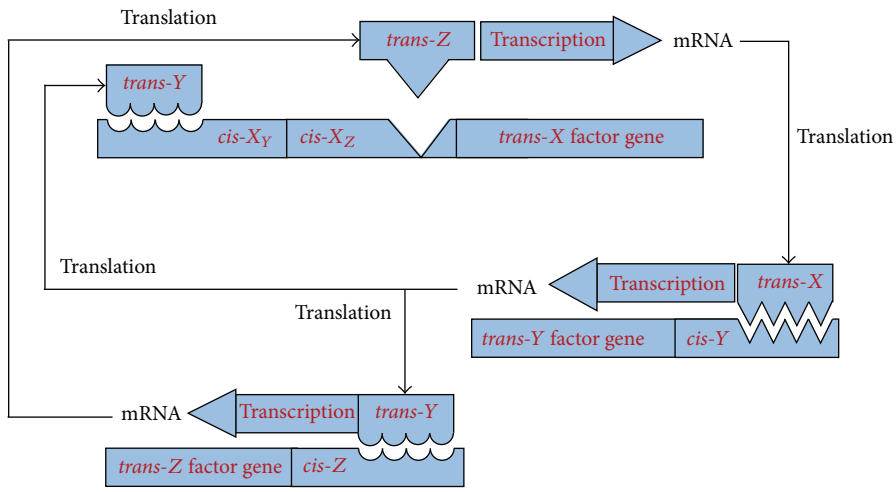

Figure 1: Gene regulation with both positive and negative feedback.

the *curse of dimensionality* [7]. Another difficulty imposed by microarray datasets is considerable noise contamination [8]. The current work deals with small to medium sized networks and hence is not faced with the former problem. However, the authors do focus on the performance of the proposed methodology in the presence of noise.

In this paper, we have proposed a new methodology for the accurate extraction of the topology of a GRN from any given noisy temporal genetic expression dataset using a statistical paradigm based on the theory of combination. The methodology has an underlying hybrid swarm intelligence framework which is basically a *Bat Algorithm* (BA) inspired *Particle Swarm Optimization* (PSO) algorithm christened BAPSO by the authors. Here, better results have been obtained compared to the contemporary literature for the benchmark networks considered. The proposed methodology uses the *Recurrent Neural Network* (RNN) for modelling the required network dynamics.

According to Bolouri and Davidson [9], a gene in a GRN is usually regulated by 4 to 8 other genes. We have proposed a novel GRN construction strategy (based on this concept) that generates candidate architectures with a limit to the maximum number of regulators for each gene in the network. Since, in this work, we have studied only small-scale and medium-scale networks, we have assumed the maximum number of regulators to be 4 [9]. The fundamental mathematical theory of combination has been applied to search all the candidate solutions in the discrete search space of network constructions exhaustively. The corresponding RNN model parameters have been trained by the proposed hybrid metaheuristic technique that can replicate the original network dynamics faithfully. The quality of a solution architecture depends on the quantum of error in the predicted dynamics. The authors have observed in this investigation that biologically plausible candidate architectures return much-reduced prediction errors compared with those which are far removed from real-world network structures.

We have implemented our proposed algorithm on three different types of data:

(i) A synthetic dataset generated from an artificial network which has been studied quite extensively concerning reverse engineering of GRNs.

(ii) A real-world experimental dataset (*in vivo*), that is, the DNA SOS repair network of *E. coli*.

(iii) An artificial dataset generated *in silico* from a real-world network of *E. coli*.

(iv) Another artificial dataset generated *in silico* from a real-world network of *yeast*.

Also, we have incorporated networks from small to medium scale in this work (i.e., 4-gene to 20-gene networks). In the case of the synthetic dataset, GRNs predicted by our proposed methodology generate improved results concerning the prediction of correct as well as incorrect regulations, compared to the best existing results in the contemporary literature (to the best of our knowledge). In the case of the *in silico* experiments, the results suggest that our proposed algorithm is robust enough to return fewer incorrect predictions along with an increase in the number of correct predictions compared to the best available outcomes in recent research endeavours. For the real experimental datasets, it has been observed that the proposed methodology can identify all the possible gene regulatory relations, some of which are quite elusive to the contemporary as well as previous researchers.

The rest of the paper has been organized as follows. The background of temporal genetic expression dataset study has been presented in the next section with an outline of the existing methodologies for reverse engineering of GRNs. The subsequent section presents our proposed framework in detail. Experimental results have been presented and discussed next. The final section concludes the paper, highlighting some future research scopes.

2. Preliminaries

2.1. Background. Traditional investigations in the domain of molecular biology provide vital information about the functioning of the genetic system in a living cell. Regrettably, this information so far is inadequate for us to comprehend the complex gene regulatory mechanisms fully. Among such techniques, southern blotting was first reported by Augenlicht and Kobrin [10], and it is the origin of DNA microarray technology [11]. Improvements in this technology have allowed us to measure the expression levels of thousands of genes simultaneously under various circumstances. These data help us in the pursuit of disclosing the knowledge about the regulatory interactions between genes of the entire genome of a living organism. Despite these advancements, there remain numerous open challenges in system biological research domain.

Various approaches exist in the contemporary literature for the construction of GRNs from time series genetic expression datasets. At the outset, researchers attempted to employ clustering algorithms on temporal expression data based on pairwise correlation coefficients [12] and Euclidian distances [13] for the reconstruction of GRNs. Information theory based approaches that made use of "mutual information" between different expression profiles have also been implemented by researchers for defining similarity between genes [14–16]. Application of Bayesian networks for modelling of GRNs is also quite popular among the researcher fraternity [17–20].

GRNs can be effectively constructed using the dynamical modelling formalisms [21] such as Boolean networks [22], where Boolean variables are used to represent the interaction between genes, and the ordinary differential equations based method, *S*-systems [23–25], where in-depth biochemical kinetic models are used to simulate gene network architectures. All of the above can reproduce the structure as well as the temporal expression profiles from temporal genetic expression profiles.

Additive regulation networks have also been used by researchers to represent the dynamics of a GRN [26]. The collective regulatory effect of a group of genes on a target gene can be represented in this formalism. The intensity and type of a particular interaction between a target (i) and a regulator (j) are defined by w_{ij}: a positive value denotes expression (facilitation) and a negative value denotes repression while a zero (0) value implies that there is no interaction between i and j. Thus, a GRN can be represented by a weight matrix $W = [w_{ij}]_{N\times N}$, where the number of genes in the GRN is equal to N [27, 28]. Another model somewhat analogous to this model is the *Recurrent Neural Network* (RNN) model, which has been effectively used in the reconstruction of GRNs from temporal expression data by several contemporary researchers [29–35]. The theoretical background of the RNN formalism has been discussed in detail in the next section. This forms the basis of our proposed modified framework. Figure 2 shows the representation of a GRN by an RNN model.

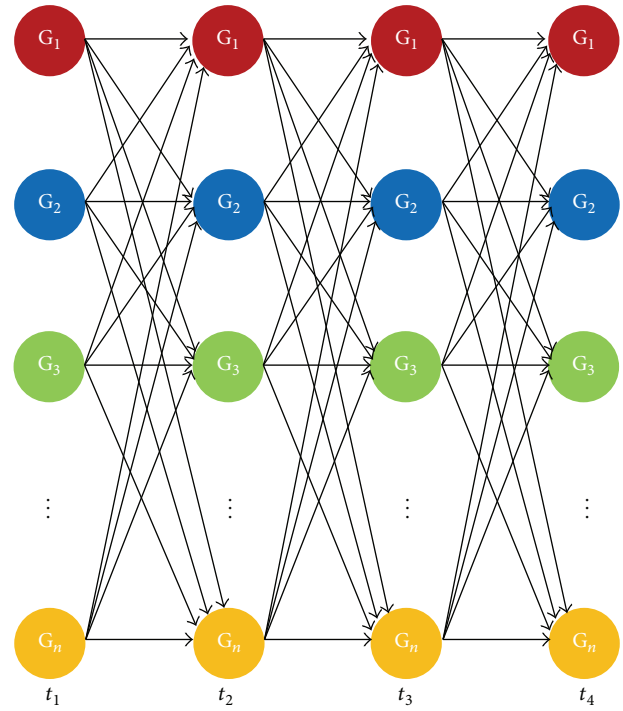

Figure 2: RNN model description of a genetic network. The network shown is unfolded from $t = t_1$ to $t = t_4$. Here, all possible connections have been shown among the genes whereas, in reality, such networks are only sparsely connected.

2.2. Recurrent Neural Networks. The regulation of the expression of any particular gene, by another gene or a group of genes, can be expressed with the help of the Recurrent Neural Network formalism [30, 36–38] as shown in Figure 2. Each node symbolises a particular gene and the edges between the nodes represent the regulatory interactions among the genes. Each tier of the neural network defines the genetic expression level of the genes at a specified time t_i. The level of expression of any particular gene at a time $t_{i+1} = t_i + dt$ depends upon the genetic expression level of all the genes (x_j) at the preceding time t_i and the weights of their corresponding connecting edges ($w_{i,j}$) with that particular gene. Thus, the total regulatory effect of all the genes in a network, on any gene i, can be summarised as follows:

$$g_i = \sum_{j=1}^{n} w_{i,j} x_j + \beta_i. \quad (1)$$

This can be transformed using a sigmoid function, within an interval $[0, 1]$, as has been shown by Vohradsky [31]. Here, β_i symbolises an external input, which may be visualised as a reaction delay parameter. A higher (large) value of this parameter indicates a reduction of the effect (influence) of $w_{i,j}$ on g_i. The actual genetic expression rate is subsequently modulated by a multiplicative constant χ_1 that defines the peak expression level of a particular gene [31]. The rate of expression of any gene i can be defined as the total of

the regulatory effects of other genes on it δ_i minus its degradation γ_i. This is represented by

$$\frac{dy_i}{dt} = \delta_i - \gamma_i, \tag{2}$$

where the degradation factor γ_i can be modelled based on the kinetic framework of a first-order biochemical equation represented as $\gamma_i = \chi_{2i} \cdot y_i$ [31]. The term δ_i represents the entire regulatory effect on the expression of the gene, i (represented as $\delta_i = \chi_{1i} \cdot f(g_i)$). The constant χ_2 signifies the rate constant of degradation of the gene product i. Thus,

$$\frac{dy_i}{dt} = \chi_{1i} \cdot f\left(\sum_{j=1}^{n} w_{i,j}x_j + \beta_i\right) - \chi_{2i}y_i, \tag{3}$$

where f denotes the sigmoid transfer function and x_j signifies the concentrations of the elements of the given system (for $j = i$; $x_j = y_i$). The above expresses the dynamics of expression of a gene i and denotes a "node" function [31]. Each node can be connected with all the other nodes to form a neural network (Figure 2). The weight matrix w describes the connection between the nodes of the network; a nonzero value of w_{ij} means that a connection between nodes i and j exists. The magnitude of the weight w_{ij} signifies the strength of an interaction, or a regulatory effect, between the two nodes. The neural network is completely defined by the differential equations respective to the particular nodes, and the number of equations is determined by the number of nodes. The quantum of genetic expression at any arbitrary time t can be calculated by solving the set of differential equations. Equation (3) represents a special case of a class of RNNs described by the more general equation:

$$\tau_i \frac{dx_i}{dt} = f_i\left(\sum_{j=1}^{n} w_{i,j}x_j + \beta_i\right) - x_i, \tag{4}$$

$$\tau_i \frac{dx_i}{dt} = \frac{1}{1 + \exp\left(\sum_{j=1}^{n} w_{i,j}x_j + \beta_i\right)} - x_i. \tag{5}$$

This is a continuous model that has been used for modelling brain activity pattern and the study of its dynamics. If the weight matrix $w_{i,j}$ is symmetrical in nature, the network it represents reaches stability in finite time. Now, in real world, time series data are obtained at discrete time points only for which (5) can be rewritten in its discrete format as follows:

$$\tau_i \frac{x_i(t + \Delta t) - x_i(t)}{\Delta t} = \frac{1}{1 + \exp\left[-\left(\sum_{j=1}^{N} w_{i,j}x_j + \beta_i\right)\right]} - x_i(t),$$

$$\text{or } x_i(t + \Delta t) = \frac{\Delta t}{\tau_i\left\{1 + \exp\left[-\left(\sum_{j=1}^{N} w_{i,j}x_j + \beta_i\right)\right]\right\}} - \left(1 - \frac{\Delta t}{\tau_i}\right)x_i. \tag{6}$$

The dynamics of a GRN can be parametrically modelled using appropriate dynamical methodologies such as Bayesian networks, Boolean networks, Recurrent Neural Networks, and S-systems. This indicates the significance of identification of the underlying information regarding genetic interactions present in the temporal expression data of a regulatory network. The purpose of any reverse engineering framework is the accurate inference of the applied model's parameters for the faithful reproduction of the given time series data. This can be viewed as an optimization problem, where the model parameters are trained to minimise the difference between the simulated and the original time series data. Determination of the *mean square error* (MSE) from the above can be a suitable measure of this specification:

$$\text{MSE} = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\left(x_i(t) - \widetilde{x}_i(t)\right)^2. \tag{7}$$

Here, N is the total number of genes (nodes) in the network, T is the total number of time points available, $x_i(t)$ is the original expression data, and $\widetilde{x}_i(t)$ is the simulated data at any point of time t.

2.3. Major Concerns. One of the major hurdles in the reverse engineering of GRN from temporal gene expression data is the *curse of dimensionality*. It arises from the fact that the number of genes in a dataset is usually two to three orders higher than the number of time points, and it severely reduces the prediction capacity of the given formalisms. Researchers have attempted to solve this problem to some extent in [28, 30, 32, 33, 39, 40]. The present work focuses on small- to medium-scale networks only (4 genes to 20 genes) and thus does not face the entire severity of this problem.

The RNN methodology has been implemented, in this paper, to model the temporal expression data. For that purpose, the RNN model parameters require training, which, in essence, is an optimization problem. Several metaheuristic techniques, like Simulated Annealing [30, 41], Genetic Algorithm (GA) [32, 33, 40, 42, 43], Differential Evolution [44], Particle Swarm Optimization [34, 45, 46], and so forth, have been and are being implemented for this purpose with various levels of accuracy. The proposed methods, however, have largely been ineffective to accurately infer even small-scale real-life GRNs. A few have been able to identify all the true regulations but in the process have also inferred unwanted false regulations. Moreover, the "No Free Lunch" (NFL) theorem [47] rationally states that there is no single metaheuristic that is most appropriate for solving all types of optimization problems. Therefore, finding out the most suitable and efficient optimization techniques for the accurate inference of small GRNs is still an open problem for researchers.

Nevertheless, the number of parameters in need of training undergoes quadratic scaling with respect to the number of genes in a GRN. This fact imposes severe hindrance in keeping the dimension of the optimization problem at a reasonable computational limit. As a result, optimization of model parameters becomes implausible for practical values of N (i.e., $N = 100, 1000$, etc.). To solve this difficulty,

researchers have proposed strategies like decomposition of the problem of global optimization of parameters into local problems of parameter optimization for a single target gene only [40, 43, 48, 49]. Other strategies, such as interpolation [36, 50, 51], the addition of noisy duplicate copies [52], and use of suitable thresholds [37, 52], have also been implemented to limit the number of optimizable parameters. Interpolation strategies usually have a drawback: they are incapable of faithfully summarising the dynamics between any two time points. According to van Someren et al., such strategies can bring about only a minimal reduction in the dimension of the optimization problem, irrespective of the number of additional time points [52]. Other strategies also fail to improve upon the situation as they also fail to add any supplementary information to the network dynamics.

Fortuitously, extensive biological research, in the perspective of reverse engineering of GRNs, confirms that there exist only a handful of genes that act as regulators in a GRN [7, 9, 19, 36, 37]; that is, GRNs are connected sparsely. Mathematically, this implies that we can assign a zero value to a large number of the model parameters that represent the one-on-one regulatory relationships. Thus, a considerable reduction in the dimensionality of the optimization problem can be achieved.

Researchers strived to develop a suitable optimization environment integrating this *sparseness concept* and achieve a significant improvement in the problem solution. This entailed some form of architectural constraint to be imposed on the predicted networks. Researchers have also found that it is possible to decouple the structural and the dynamic aspects of the given reverse engineering problem. In other words, there is scope for the application of a suitable technique that can decouple the problem into two independent subproblems: the search for candidate architectures in the discrete search space of network structures and the search for suitable model parameters in the corresponding continuous search space of parameters of dynamical formalisms [34, 35, 45, 46, 53, 54]. Thus, an endeavour to locate suitable model parameters may supervise the pursuit of detection of the candidate networks. The accuracy of a trained model, assessed from the perspective of reproducing the original dynamics, determines the appropriateness of a predicted architecture. The level of precision can be ascertained from the MSE calculated for the predicted models as per (7). A genetic interaction is appended to a predicted architecture or removed from it based on the value of the calculated MSE. Thus, the extra burden of searching the biologically plausible network architectures within a separate discrete search space of candidate network architectures is compensated by a considerable reduction in the dimension of the problem of training the dynamic model parameters.

3. Methods

3.1. Decoupled Strategy. In this section, we have explained in detail the methodology implemented in this work. Firstly, we have represented a GRN with the help of a directed graph; $G = (V, E)$ represents a GRN, where V denotes the set of all nodes (*genes*) and E is the set of all edges (*the interaction between a pair of genes*). An edge, $e_{i,j}$, is present in the set E if and only if there exists an interaction between node (*gene*) i and node (*gene*) j. Here, $e_{i,j}$ signifies that gene j regulates gene i, and this convention has been used right through this work. A directed graph can be represented by an adjacency matrix for computational purposes. An adjacency matrix $G = [g_{i,j}]_{N\times N}$, where N is the number of nodes in the graph (i.e., *the number of genes in the network*). The element $g_{i,j}$ can take any value, 0 or 1, depending on the absence or presence of a directed edge from node j to node i, respectively.

Now, the methodology proposed in this work, for the reverse engineering of GRNs from temporal expression datasets, employs the decoupling strategy discussed in the previous section [34, 35, 45, 46, 53, 54]. Here, we have first reduced the search space of candidate network structures by restricting the number of regulators [9] on a particular gene in a GRN. Subsequently, we have implemented the theory of combination to exhaustively search the reduced candidate network architecture space. In other words, if there are N genes in a GRN and m is the maximum number of regulators allowed for a gene, then the search space dimension is, by definition, ${}^{N}C_{m}$ or $\binom{N}{m}$. This is much less than the original search space dimension of 2^N. Additionally, since our proposed algorithm is performing exhaustive search in the reduced space, it has a high chance of obtaining biologically plausible candidate network architectures. Mathematically, there are ${}^{N}C_{m}$ GRN structures, each represented by G_i ($i = 1, 2, \ldots, {}^{N}C_{m}$).

In the next phase, the RNN formalism has been implemented to model the underlying dynamics from the temporal genetic expression profiles based on the candidate network structures obtained in the previous phase. In other words, the weight matrix W_i of the RNN formalism has been initialised based on all G_i's defined. We have used the proposed BAPSO technique to train the RNN model parameters, that is, w_{ij}, β_i, and τ_i, accurately such that the predicted expression profiles match the original expression profiles faithfully. The MSE defined by (7) determines the quality of a candidate solution G_i, and the candidate solution with the least MSE has been chosen as the most reasonable from all the ${}^{N}C_{m}$ or $\binom{N}{m}$ candidates.

It would be interesting to note here that each of the genes in a GRN may not always be regulated by the maximum number of allowed regulators; that is, $m = 4$ genes. Therefore, we have gradually incremented the value of m from 1 to 4, and the MSE has been calculated for each case. A satisfactorily low value of MSE $\sim 10^{-3}$ has been used as the selection criterion for a candidate solution.

A further problem encountered in this endeavour is the dimensionality of the RNN model parameter training problem. For N genes in a GRN, there are $N \times (N + 2)$ parameters to be trained for a particular RNN instance with the help of the BAPSO technique, and this essentially becomes computationally unrealistic for large values of N. To further reduce the computational load, in this work, we have decomposed this problem into N subproblems where, in each subproblem, $(N + 2)$ parameters are trained for each of the N genes, independently. In case of each of

the independent subproblems, the aim is to minimise the error term er_i defined as

$$\mathrm{er}_i = \frac{1}{T}\sum_{t=1}^{T}\left(x_i(t) - \widetilde{x}_i(t)\right)^2. \tag{8}$$

Here, $\mathrm{er}_i \in E$ and $E = [\mathrm{er}_i]_{1\times N}$ which is subsequently used for the calculation of MSE. Hence,

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}\mathrm{er}_i. \tag{9}$$

The MSE governs the overall quality of the candidate solutions. The lower the value of the term er_i, the more efficient the reduction in the difference between the predicted temporal expression profile and the original one and the more suitable the candidate network architecture. It is, therefore, the ultimate objective of the proposed methodology to reconstruct a network architecture that is biologically plausible and at the same time capable of replicating the original network dynamics more accurately.

3.2. The Proposed Metaheuristic. The training of the model parameters of the RNN instances has been achieved using the proposed BAPSO algorithm. Among all the proposed swarm intelligence techniques to date, Particle Swarm Optimization (PSO) [55–58] is conspicuous for being simple yet efficient, robust, easily tractable, and easy to code. PSO yields solutions that are of the same or better quality compared to GA for a wide array of problems and possesses a faster convergence rate. A particle swarm comprises some particles arbitrarily dispersed in a search space. The positions of these individual particles denote candidate solutions. The intention of any particle is to find the optimum solution utilising the knowledge acquired through social interactions with its neighbours. Each particle in a swarm is specified by its position p_{pso}, its velocity v_{pso}, and its memory of the best solution achieved by it so far p^b_{pso}. Another memory element g^b denotes the best solution attained thus far by the swarm and is shared among all particles.

The position of a particle signifies the vector containing all the parameters of an RNN instance. The fitness of a particle is calculated using either (8) or (7), depending upon whether the decoupled strategy has been implemented or not, respectively. In other words, if someone chooses not to use the decoupled strategy, then the quality of the solution is determined by (7). However, since, in the decoupled strategy, each gene is dealt with separately, the quality of a solution is determined by (8), and we have used this only. For each generation, the updated position p'_{pso} and velocity v'_{pso} of a particle for the next generation are calculated based on its best solution achieved so far and the best solution obtained by the entire swarm thus far. Hence,

$$\begin{aligned} v'_{\mathrm{pso}_i} = {} & w \otimes v_{\mathrm{pso}_i} + r_1c_1 \otimes \left(p^b_{\mathrm{pso}_i} - p_{\mathrm{pso}_i}\right) + r_2c_2 \\ & \otimes \left(g^b - p_{\mathrm{pso}_i}\right), \end{aligned} \tag{10}$$

$$p'_{\mathrm{pso}_i} = p_{\mathrm{pso}_i} + v'_{\mathrm{pso}_i}, \tag{11}$$

where w is the inertia weight term, and it controls the dynamic balance between exploration and exploitation undertaken by a particle. Again, r_1 and r_2 are random numbers in the range [0, 1] and usually $c_1 = c_2 = 2$. The terms r_1c_1 and r_2c_2 determine the effect (on the particle velocity) of the best solutions achieved by a particle and the swarm, respectively. The terms are all in a matrix format and thus it is sensible to point out that elementwise multiplications are necessary here and have been symbolised by ⊗.

BA has been recently formulated by Yang based on the echolocation property of real bats [59, 60]. In BA, the virtual bats locate food and inform others about the food source with the help of sound waves. The virtual bats are assumed to have the ability to modulate the sound waves according to the need, that is, locating food/prey or communicating with others. The virtual bats are also scattered in the search space, with the position of each virtual bat denoting possible solutions. A virtual bat is completely specified by its position p_{ba}, its velocity v_{ba}, loudness A, and frequency f. A memory element p_{best} stores the position of the best food source found so far. The velocity and position of a virtual bat are updated according to the following equations:

$$\begin{aligned} f_i &= f_{\min} + \mu \otimes \left(f_{\max} - f_{\min}\right), \\ v'_{\mathrm{ba}_i} &= v_{\mathrm{ba}_i} + f_i \otimes \left(p_{\mathrm{best}} - p_{\mathrm{ba}_i}\right), \\ p'_{\mathrm{ba}_i} &= p_{\mathrm{ba}_i} + v'_{\mathrm{ba}_i}, \end{aligned} \tag{12}$$

where $\mu \in [0, 1]$ is a random vector. In this investigation, if standalone BA had been used, then the pertinent values would have been $f_{\min} = 0$ and $f_{\max} = 1$. At the outset, each virtual bat is arbitrarily allocated a frequency from $[f_{\min}, f_{\max}]$, drawn uniformly. This frequency term controls the movement of the virtual bats in the search space, similar to what the inertia weight term does in case of PSO, as can be seen in (10). There are various ways of updating the inertia weight for PSO.

In this paper, we have proposed a new technique based on the update technique of frequency of virtual bats in BA. We have proposed to update the inertia weight w in PSO in each iteration using the following equation:

$$w_i = w_{\min} + \mu \otimes \left(w_{\max} - w_{\min}\right), \tag{13}$$

where $\mu \in [0, 1]$ is also a random vector. We have assumed $w_{\min} = 0$ and $w_{\max} = 1$. In each iteration, for each particle, an inertia weight is drawn uniformly from $[w_{\min}, w_{\max}]$. This somewhat counterbalances the problem of being trapped at local minima, which is one of the few but major shortcomings of PSO. The proposed novel BAPSO algorithm, with the new inertia weight update technique, used for the particular problem domain dealt with herein, has been able to produce better results than individual PSO or BA algorithms (as suggested by other investigations carried out by the authors).

Another change, inspired by the virtual bats in BA that has been incorporated in the proposed BAPSO algorithm, is the initialization of the velocity vector of each particle to zero instead of a random vector. The authors observe that this might help in preventing the particles from having an initial

unguided velocity that may divert them away from a potential optimal solution in the search space.

4. Experimental Results and Discussion

Owing to the stochastic nature of the proposed framework implemented, it is quite normal that, for any given temporal genetic expression dataset, the predicted GRN would vary in its topology for each independent solution generated. To circumvent this problem, we have employed, in this investigation, a collaborative learning method. We have performed L independent experiments and have stored in memory each L inferred GRN. Additionally, a selection scheme has been implemented based on a plausibility score $\mathrm{ps}_{i,j}$, assigned to each edge $e_{i,j}$, as given below. This has been done to identify the most consistent predicted edges for the construction of the final GRN:

$$\mathrm{ps}_{i,j} = \frac{1}{L}\sum_{1}^{L} g_{i,j}. \tag{14}$$

In (14), $g_{i,j} \in G$ and $\mathrm{ps}_{i,j} \in [0, 1]$. On the evaluation of $\mathrm{ps}_{i,j}$ for all i and j, the final predicted network, thus, can be generated and represented by $G_F = [g^f_{i,j}]_{N\times N}$. Whether the value of a particular element $g^f_{i,j}$ is 0 or 1 can be evaluated using the following relation:

$$g^f_{i,j} = \begin{cases} 1, & \text{if } \mathrm{ps}_{i,j} \geq \alpha, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

In the above equation, α is a threshold defined for the purpose of inclusion of an interaction in a GRN. In other words, it governs whether an edge is included in G_F or omitted altogether. In order to estimate the accuracy of the proposed methodology, we have compared G_F with the original GRN, denoted by G_O. In addition, we have quantitatively compared the results of the proposed framework with those from the contemporary literature based on certain metrics. Before explaining the metrics, it would be prudent to mention that an edge can be characterised into four types: true positive (TP), false positive (FP), true negative (TN), and false negative (FN), with their mathematical definitions as follows:

$$\text{TP: if } g^o_{i,j} = 1 \text{ and } g^f_{i,j} = 1; \text{ TN: if } g^o_{i,j} = 0 \text{ and } g^f_{i,j} = 0.$$

$$\text{FP: if } g^o_{i,j} = 0 \text{ and } g^f_{i,j} = 1; \text{ FN: if } g^o_{i,j} = 1 \text{ and } g^f_{i,j} = 0.$$

Next, we have defined the metrics one by one based on which the proposed methodology can be evaluated.

(i) True Positive Rate (TPR)/Sensitivity/Recall. This signifies the fraction of the total number of existing edges in the original network, correctly predicted in the inferred network.

(ii) True Negative Rate/Specificity (SPC). This signifies the fraction of the total number of nonexistent edges in the original network, correctly identified as nonexistent in the inferred network as well.

(iii) False Positive Rate (FPR)/Complimentary Specificity. This signifies the fraction of the total number of nonexistent edges, incorrectly predicted in the inferred network.

(iv) False Negative Rate (FNR)/Complimentary Sensitivity. This signifies the fraction of the total number of nonexistent edges in the original network, incorrectly guessed in the predicted network.

(v) Positive Predictive Value (PPV)/Precision. This signifies the fraction of the total number of inferred edges, which is correct.

(vi) False Discovery Rate (FDR)/Complimentary Precision. This signifies the fraction of the total number of inferred edges, which is incorrect.

(vii) Accuracy (ACC). This signifies the fraction of the total number of all possible connections, in the original network, truly predicted.

(viii) F-Score. This signifies the harmonic mean of the precision and sensitivity.

Mathematically speaking,

$$\begin{aligned}
\mathrm{TPR} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \\
\mathrm{SPC} &= \frac{\mathrm{TN}}{\mathrm{FP}+\mathrm{TN}}, \\
\mathrm{FPR} &= \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}} = 1 - \mathrm{SPC}, \\
\mathrm{FNR} &= \frac{\mathrm{FN}}{\mathrm{TP}+\mathrm{FN}} = 1 - \mathrm{TPR}, \\
\mathrm{PPV} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \\
\mathrm{FDR} &= \frac{\mathrm{FP}}{\mathrm{TP}+\mathrm{FP}} = 1 - \mathrm{PPV}, \\
\mathrm{ACC} &= \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{FP}+\mathrm{FN}+\mathrm{TN}}, \\
F &= \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}.
\end{aligned} \tag{16}$$

The statistical BAPSO methodology has been applied primarily on an artificial network (4 genes). Subsequently, we have applied the proposed methodology on a group of experimental (*in vivo*) time series genetic expression datasets of a real-world network (the 8-gene *E. coli* SOS DNA repair network). Finally, we have experimented with two networks extracted from the genome of *Saccharomyces cerevisiae* (10 genes) and *Escherichia coli* (20 genes) with the help of GeneNetWeaver [63]. Additionally, we have implemented our proposed algorithm on each of these networks, but with half the number of time points initially used for experimentation. This has been done to observe the accuracy of the method if a lesser number of time points are available for training.

Table 1: RNN model parameters [32, 34].

$w_{i,j}$				b_i	c_i
20	−20	0	0	0	10
15	−10	0	0	−5	5
0	−8	12	0	0	5
0	0	8	−12	0	5

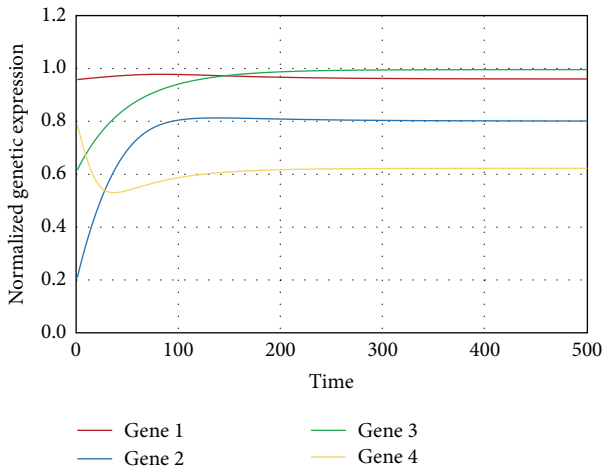

Figure 3: Network dynamics used for training the proposed model. The four lines represent the expression profile of the four genes.

All the simulations have been run on a desktop computer running on a 3.4 GHz Intel Core i7 processor with 8 GB 1600 MHz RAM. The codes have been run on Matlab 2014a, running in a Windows 7 64-bit environment.

4.1. Artificial Network. This artificial network consists of 4 genes and 8 interactions. This network has been extensively studied by researchers for the purpose of preliminary validation of their methodologies with respect to reverse engineering of GRNs from time series genetic microarray data [32, 34, 46]. The time series expression data have been generated using (3). The parameters and their related values necessary for calculations have been given in Table 1. The generated expression profiles have been shown in Figure 3. We have assumed $\Delta t = 0.1$ for this case and have generated 500 time points with the help of (3).

However, in real-world experiments, such a high number of time points do not typically exist. Therefore, we have sampled the data evenly into 50 time points and have implemented our proposed methodology on the sampled dataset. Further, we have evenly sampled this reduced dataset to produce another dataset with 25 time points.

The reverse engineering initiative involves $L = 10$ independent experiments. We have conducted each experiment with a swarm population of ${}^4C_m$ (where $m = 1, 2, 3, 4$) particles and 100000 iterations. The statistical properties of the final inferred network have been shown in Figure 4, for $\alpha = 0.9$. Utilising just a single time series, the results show marked improvement over those published by Xu et al. [34] and Kentzoglanakis and Poole [46], with respect to both true

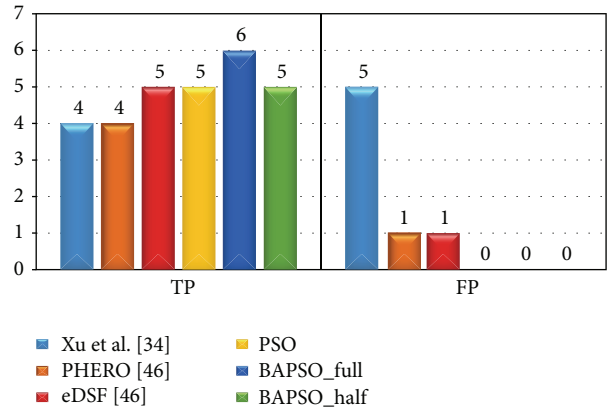

Figure 4: True positive (TP) and false positive (FP) counts obtained by the proposed BAPSO model, compared with those obtained by Xu et al. [34] and Kentzoglanakis and Poole [46] and PSO. The results of the BAPSO model have been presented for two datasets: one with 50 time points, represented as BAPSO_full, and the other with 25 time points, represented as BAPSO_half.

and false positives albeit with a stricter threshold than that used by the authors ($\sigma = 0.5$) in [46]. The average MSE for the experiments are $\sim 10^{-6}$ and $\sim 10^{-5}$ for the dataset with 50 time points and the one with 25 time points, respectively. The computational times for both experiments are 15.6 minutes and 8.6 minutes, respectively.

4.2. E. coli DNA SOS Repair Network. In this section, the proposed methodology for reverse engineering of GRNs from temporal genetic expression profiles has been employed to identify the causal relationships among the genes from an *in vivo* (experimental) microarray dataset. The said dataset summarises the dynamics of the well-illustrated transcriptional network involved in the SOS DNA repair mechanism of *E. coli* studied experimentally by Ronen et al. [64]. The study included eight genes heavily involved in the SOS repair mechanism: recA, lexA (the master repressor), uvrA, uvrD, uvrY, umuD, ruvA, and polB. The original network has been shown in Figure 5. Four experimental datasets had been generated using two different UV light intensities on *E. coli* (for experiments 1 and 2: UV intensity used $\rightarrow 20\,\mathrm{Jm}^{-2}$; for experiments 3 and 4: UV intensity used $\rightarrow 5\,\mathrm{Jm}^{-2}$). In each of the experiments, expression data had been observed for 50 time points each using temporal resolution of 6 minutes. These datasets are one of the most useful ones concerning the qualitative investigations on computational methods for reconstruction of GRNs from time series genetic expression data (which for ready reference is at http://wws.weizmann.ac.il/mcb/UriAlon/sites/mcb.UriAlon/files/uploads/DownloadableData/sosdata.zip).

In this case also, $L = 10$ independent experiments have been performed for each of the four datasets. A swarm population of ${}^8C_m$ (where $m = 1, 2, 3, 4$) has been used with a maximum number of iterations set to 5000. The expression value of each gene in each dataset at the first time

Table 2: Results of the *E. coli* experiments with all the available time points.

Dataset	TP	TN	FP	FN	TPR	SPC	FPR	FNR	PPV	FDR	ACC	*F*1-score	Graph edges	MSE	CPU time
BAPSO_full															
1	7	46	9	2	0.78	0.84	0.16	0.22	0.44	0.56	0.83	0.56	16	0.0157	16.9 m
2	7	40	15	2	0.78	0.73	0.27	0.22	0.32	0.68	0.73	0.45	22	0.0114	16.9 m
3	7	45	10	2	0.78	0.82	0.18	0.22	0.41	0.59	0.81	0.54	17	0.0147	16.9 m
4	4	43	12	5	0.44	0.78	0.22	0.56	0.25	0.75	0.73	0.32	16	0.0147	16.9 m
BAPSO_half															
1	7	38	17	2	0.78	0.69	0.31	0.22	0.29	0.71	0.70	0.42	24	0.0042	9.1 m
2	8	40	15	1	0.89	0.73	0.27	0.11	0.35	0.65	0.75	0.50	23	0.0025	9.2 m
3	6	43	12	3	0.67	0.78	0.22	0.33	0.33	0.67	0.77	0.44	18	0.0048	9.1 m
4	4	40	15	5	0.44	0.73	0.27	0.56	0.21	0.79	0.69	0.29	19	0.0039	9.1 m

Table 3: Comparison of results obtained from the *E. coli* experiments with [46].

Dataset	TP				FP			
	eDSF [46]	PSO	BAPSO_full	BAPSO_half	eDSF [46]	PSO	BAPSO_full	BAPSO_half
1	3	5	**7**	**7**	10	9	**9**	**17**
2	8	4	**7**	**8**	5	10	**15**	**15**
3	4	5	**7**	**6**	9	8	**10**	**12**
4	0	3	**4**	**4**	9	8	**12**	**15**

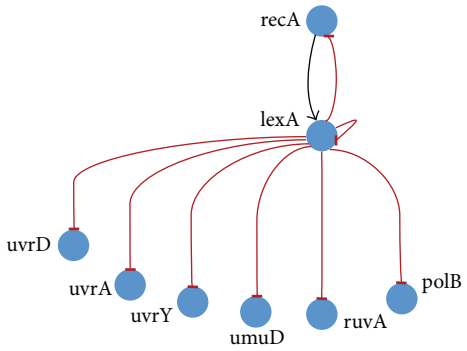

Figure 5: The original structure of the SOS DNA repair transcriptional network of *E. coli*.

Table 4: Comparison with contemporary research [46] for the *E. coli* experiments.

Known interactions	Predictions by						
	[13]	[61]	[34]	[46]	[62]	PSO	BAPSO
lexA → lexA	Yes	Yes	No	Yes	Yes	Yes	Yes
lexA → recA	Yes	Yes	Yes	Yes	Yes	Yes	Yes
recA → lexA	Yes	Yes	No	No	No	Yes	Yes
lexA → uvrA	Yes	Yes	Yes	Yes	Yes	Yes	Yes
lexA → uvrD	No	No	Yes	Yes	Yes	Yes	Yes
lexA → uvrY	No	No	No	Yes	No	Yes	Yes
lexA → umuD	No	Yes	Yes	Yes	Yes	Yes	Yes
lexA → ruvA	No	No	No	Yes	No	Yes	Yes
lexA → polB	No	No	Yes	Yes	Yes	Yes	Yes
Spurious edges (FP)	5	10	2	5	3	10	9
Precision (PPV)	0.44	0.33	0.71	0.62	0.70	0.47	0.44

point is zero and hence has been ignored. Subsequently, all expression values have been normalised to the range $[0, 1]$. The dataset thus contains 49 time points. We have also taken alternative time points and created a truncated dataset with 25 points. The statistical properties of the predicted GRN in each experiment with a plausibility score threshold, set at $\alpha = 0.9$, have been shown in Table 2.

Table 3 displays a quantitative comparison of the characteristics of the predicted GRNs (with a plausibility score threshold, $\alpha = 0.9$) with those presented in a recent investigative work (with an inclusion threshold, $\sigma = 0.9$) [46]. The proposed methodology is consistent regarding the number of true (and false) positives predicted compared to results presented in [46] for different experimental datasets. The method proposed in [46] fails to identify any true positive in the fourth experiment whereas the framework proposed in this paper does not fail to identify true positives for any experiment. However, we have to concede that the proposed framework cannot match the isolated best result obtained by the eDSF model [46] in the case of the second experiment. However, it may be noted that the regulatory relationship between recA and lexA was not inferred in any of the experiments conducted by Kentzoglanakis and Poole [46], whereas the proposed methodology can identify this particular interaction in one of the four experiments. This suggests that it probably is among a few proposed computational frameworks that are capable of identifying all the regulatory interactions present in the SOS response network of *E. coli*. A qualitative comparison of several such methodologies [19, 34, 46, 61, 62] is given in Table 4. The performance of the methodology with half the number of time points is also admirable and has been included in Tables 2 and 3.

Table 5: Results for the yeast dataset extracted from GNW with 50 time points and 25 time points represented as BAPSO_full and BAPSO_half.

α	TP	TN	FP	FN	TPR	SPC	FPR	FNR	PPV	FDR	ACC	$F1$-score	Graph edges	MSE	CPU time
BAPSO_full															
0.5	6	75	13	6	0.50	0.85	0.15	0.50	0.32	0.68	0.81	0.39	19	0.0034	27.4 minutes
0.6	6	75	13	6	0.50	0.85	0.15	0.50	0.32	0.68	0.81	0.39	19		
0.7	6	76	12	6	0.50	0.86	0.14	0.50	0.33	0.67	0.82	0.40	18		
0.8	6	76	12	6	0.50	0.86	0.14	0.50	0.33	0.67	0.82	0.40	18		
0.9	5	78	10	7	0.42	0.89	0.11	0.58	0.33	0.67	0.83	0.37	15		
1.0	5	79	9	7	0.42	0.90	0.10	0.58	0.36	0.64	0.84	0.38	14		
BAPSO_half															
0.5	4	73	15	8	0.33	0.83	0.17	0.67	0.21	0.79	0.77	0.26	19	0.0048	14.7 minutes
0.6	4	73	15	8	0.33	0.83	0.17	0.67	0.21	0.79	0.77	0.26	19		
0.7	4	74	14	8	0.33	0.84	0.16	0.67	0.22	0.78	0.78	0.27	18		
0.8	4	74	14	8	0.33	0.84	0.16	0.67	0.22	0.78	0.78	0.27	18		
0.9	4	76	12	8	0.33	0.86	0.14	0.67	0.25	0.75	0.80	0.29	16		
1.0	4	77	11	8	0.33	0.88	0.13	0.67	0.27	0.73	0.81	0.30	15		

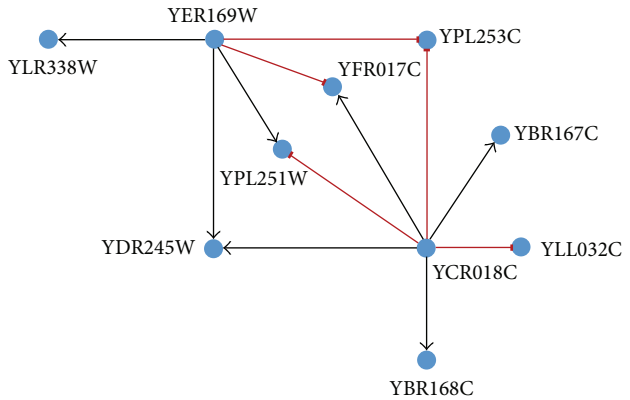

Figure 6: Network architecture extracted from GNW to validate the proposed framework as used in [46].

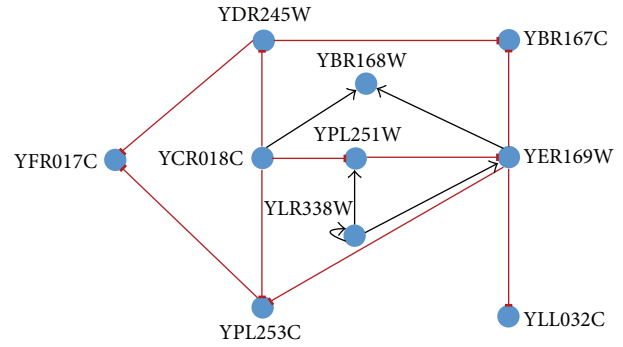

Figure 7: Inferred network obtained by the proposed model for 50 time points.

4.3. 10-Gene Network Extracted from GeneNetWeaver (GNW). The *in silico* datasets have been extracted from the genome of yeast and *E. coli* stored in GNW [63]. First, we have considered the yeast network, made up of 10 genes and 12 genetic interactions as shown Figure 6. We have generated the network dynamics with the help of GeneNetWeaver [63] in keeping with DREAM4 settings [65]. Two sets of genetic expression data have been generated, one with 50 time points and the other with 25 time points (taking the alternate time points of the former). The number of independently generated solutions is $L = 10$. Since there are 10 genes in the GRN, the problem has been divided into 10 subproblems, each with 12 parameters to optimise. For each of the suboptimization problems, a swarm population of ${}^{10}C_m$ (where $m = 1, 2, 3, 4$) has been used, and the maximum number of iterations has been set to 10000.

The results achieved in this experiment have been summarised in Table 5. The proposed methodology can correctly predict 5 (for $\alpha \geq 0.9$) out of a possible 12 interactions present in the original network using the dataset with 50 time points. The proposed methodology can also correctly predict 4 (for $\alpha \geq 0.9$) out of a possible 12 interactions present in the original network using the dataset with 21 time points. With the increase in α, the number of incorrect predictions goes down from 13 to 9, increasing the accuracy from 81% to 84%, and from 15 to 11, increasing the accuracy from 77% to 81%, respectively, in the two cases. The final predicted GRNs for the two cases have been shown in Figures 7 and 8.

The results have been compared with previous similar work published in [46] and have been shown in Table 6. The proposed methodology indicates improvement from the perspective of true predictions. Even for the most stringent value of the threshold, that is, $\alpha = 1$, the number of true predictions is significantly more (5 compared to 3) with 50 time points and still better (4 compared to 3) with 25 time points. The true positive rate and the precision of the predicted network are almost always better than the compared network. Considering the nature of the inferred relationships (whether activation or repression), the proposed methodology has correctly identified the nature of 80% of the predicted relationships.

Table 6: Comparison of BAPSO results for the GNW dataset of yeast with PSO and eDSF [46].

Threshold	PSO	BAPSO_full	BAPSO_half	eDSF [46]	PSO	BAPSO_full	BAPSO_half	eDSF [46]
			TPR				FPR	
0.5	0.42	**0.50**	0.33	0.50	0.14	**0.15**	0.17	0.19
0.6	0.42	**0.50**	0.33	0.42	0.13	**0.15**	0.17	0.14
0.7	0.42	**0.50**	0.33	0.42	0.08	**0.14**	0.16	0.14
0.8	0.42	**0.50**	0.33	0.42	0.08	**0.14**	0.16	0.15
0.9	0.42	**0.42**	0.33	0.42	0.06	**0.11**	0.14	0.13
1.0	0.42	**0.42**	0.33	0.25	0.05	**0.10**	0.13	0.08
			Accuracy				Graph edges	
0.5	0.81	**0.81**	0.77		17	**17**	19	19
0.6	0.82	**0.81**	0.77		16	**16**	19	19
0.7	0.86	**0.82**	0.78	Not available	12	**12**	18	18
0.8	0.86	**0.82**	0.78		12	**12**	18	18
0.9	0.88	**0.83**	0.80		10	**10**	16	15
1.0	0.89	**0.84**	0.81		9	**9**	15	14

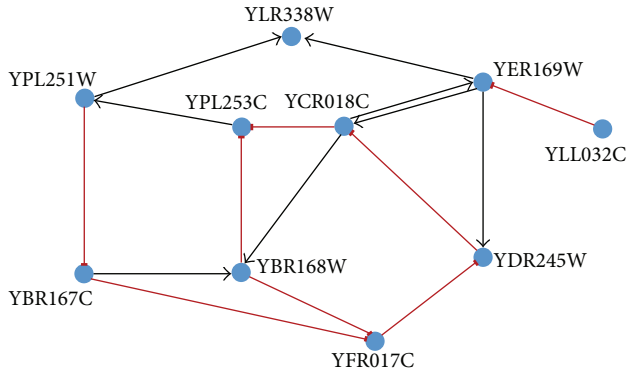

Figure 8: Inferred network obtained by the proposed model for 25 time points.

4.4. 20-Gene Network Extracted from GeneNetWeaver (GNW). The second network extracted from GNW is a 20-gene network consisting of 24 interactions. The datasets for this network have been generated using the same settings as the previous one. We have generated $L = 10$ independent solutions. There are 20 genes in this GRN, and hence the problem has been divided into 20 subproblems, each with 22 parameters to optimise. For each of the suboptimization problems, a swarm population of ${}^{20}C_m$ (where $m = 1, 2, 3, 4$) has been used, and the maximum number of iterations has been set to 10000. The original network is shown in Figure 9.

The proposed RNN based framework does not scale up with the size of the GRN, efficiently. For the 20-gene network considered here, it was able to predict only 3 out of a possible 24 interactions correctly, but with a large number of false positives. Fascinatingly, however, the proposed method can correctly predict 5 interactions out of a possible 24, with 2 less false positives. The predicted correct relations have been shown in Table 7.

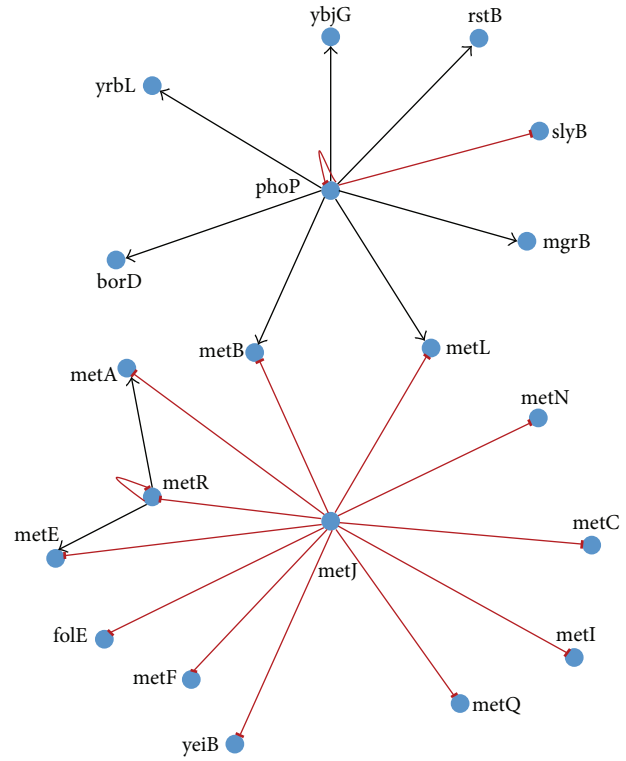

Figure 9: Original 20-gene network extracted from the genome of *E. coli* stored in GNW.

5. Conclusion

In this paper, we have investigated the domain of reconstruction of GRNs from time series microarray datasets with modifications in the existing methodologies. For this purpose, we have implemented a decoupled technique based on the novel BAPSO algorithm, the fundamental mathematical

Table 7: True positives obtained for the GRN consisting of 20 genes.

Technique	BAPSO_full	BAPSO_half
Correct interactions	metJ → metN, metJ → folE, and metR → metL	metJ → metN, metJ → metC, metJ → metQ, phoP → yrbL, and phoP → borD
Computational time	3.2 hrs	1.7 hrs
MSE	0.0031	0.0034

theory of combination, and RNN. The main objective of the investigation is to detect the biologically relevant GRNs from the large discrete network architecture search space. Prior knowledge and the fundamental theory of combination have been used for the purpose of reducing the dimension of the optimisation problem, thus reducing the computational load. Also, the proposed methodology ensures a higher probability of identifying a more biologically relevant network as it searches all possible candidate architectures (i.e., all possible combinations).

The proposed novel hybrid swarm intelligence scheme, BAPSO, has been implemented in the present investigation to train the RNN model parameters and the results obtained show that the predicted networks reproduce the dynamics of the given dataset to a better extent for small-scale GRNs. The results suggest that the proposed decoupled reverse engineering approach is robust and consistent with respect to the number of correct and incorrect predictions while using different types of microarray datasets (synthetic, *in silico*, and *in vivo*) for most of the small-scale GRNs studied in the contemporary literature.

However, it is an entirely different scenario for medium-scale networks (20 genes). The methodology fails to reproduce any of the successes it had against smaller GRNs. There are too few true predictions and a large number of incorrect predictions. The methodology, implemented in this paper, thus, needs to be enriched further by studying its performance in larger networks. This provides a vital scope for further research.

Also, the assumption of the value of the threshold, α, is based on the knowledge of the final network to be obtained. In real-world cases, where the final GRN is not known, the setting of a suitable threshold for the ensemble learning scheme used in this work needs further research. Additionally, the reduction in false positives is also an important research endeavour for the future.

Another point to be noted in this context is the performance of the methodology with a lesser number of time points, half to be exact. The results indicate that a 50% reduction in the number of time points leads to only a small drop in accuracy of the predicted models, a maximum of just over 15% in the worst-case scenario. However, interestingly, the methodology slightly improves upon the poor results obtained for the 20-gene GRN, with a lesser number of time points available.

This also provides an opportunity for future research into the prediction of GRNs from genetic expression profiles with lesser time points and will surely help to reduce time and cost of data generation in the future.

Disclosure

Appropriate references have been given in the text; no figure or table has been reproduced without permission, and none of the results has been fabricated, to the best of the authors' knowledge.

Competing Interests

The authors declare that they have no competing interests.

Authors' Contributions

All authors have equally contributed to this work.

References

[1] G. J. McLachlan, K.-A. Do, and C. Ambroise, *Analyzing Microarray Gene Expression Data*, vol. 422, John Wiley & Sons, New York, NY, USA, 2005.

[2] R. Xu and D. Wunsch II, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

[3] Z. Bar-Joseph, "Analyzing time series gene expression data," *Bioinformatics*, vol. 20, no. 16, pp. 2493–2503, 2004.

[4] P. D'haeseleer, X. Wen, S. Fuhrman, and R. Somogyi, "Linear modelling of mRNA expression levels during CNS development and injury," in *Pacific Symposium on Biocomputing*, vol. 4, no. 1, pp. 41–52, 1999.

[5] H. De Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann, "Qualitative simulation of genetic regulatory networks using piecewise-linear models," *The Bulletin of Mathematical Biology*, vol. 66, no. 2, pp. 301–340, 2004.

[6] I. M. Ong, J. D. Glasner, and D. Page, "Modelling regulatory pathways in *E. coli* from time series expression profiles," *Bioinformatics*, vol. 18, supplement 1, pp. S241–S248, 2002.

[7] D. L. Donoho, "High-dimensional data analysis: the curses and blessings of dimensionality," *AMS Math Challenges Lecture*, pp. 1–32, 2000.

[8] E. P. van Someren, L. F. A. Wessels, E. Backer, and M. J. T. Reinders, "Genetic network modeling," *Pharmacogenomics*, vol. 3, no. 4, pp. 507–525, 2002.

[9] H. Bolouri and E. H. Davidson, "Modeling transcriptional regulatory networks," *BioEssays*, vol. 24, no. 12, pp. 1118–1129, 2002.

[10] L. H. Augenlicht and D. Kobrin, "Cloning and screening of sequences expressed in a mouse colon tumour," *Cancer Research*, vol. 42, no. 3, pp. 1088–1093, 1982.

[11] E. M. Southern, "Detection of specific sequences among DNA fragments separated by gel electrophoresis," *Journal of Molecular Biology*, vol. 98, no. 3, pp. 503–517, 1975.

[12] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 95, no. 25, pp. 14863–14868, 1998.

[13] X. Wen, S. Fuhrman, G. S. Michaels et al., "Large-scale temporal gene expression mapping of central nervous system development," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 95, no. 1, pp. 334–339, 1998.

[14] A. J. Butte and I. S. Kohane, "Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements," in *Proceedings of the Pacific Symposium on Biocomputing (PSB '00)*, vol. 5, pp. 418–429, January 2000.

[15] K. Basso, A. A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera, and A. Califano, "Reverse engineering of regulatory networks in human B cells," *Nature Genetics*, vol. 37, no. 4, pp. 382–390, 2005.

[16] A. A. Margolin, I. Nemenman, K. Basso et al., "ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context," *BMC Bioinformatics*, vol. 7, supplement 1, article S7, 2006.

[17] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian networks to analyse expression data," *Journal of Computational Biology*, vol. 7, no. 3-4, pp. 601–620, 2000.

[18] D. Husmeier, "Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks," *Bioinformatics*, vol. 19, no. 17, pp. 2271–2282, 2003.

[19] B.-E. Perrin, L. Ralaivola, A. Mazurie, S. Bottani, J. Mallet, and F. D'Alché-Buc, "Gene networks inference using dynamic Bayesian networks," *Bioinformatics*, vol. 19, supplement 2, pp. ii138–ii148, 2003.

[20] I. Pournara and L. Wernisch, "Reconstruction of gene networks using Bayesian learning and manipulation experiments," *Bioinformatics*, vol. 20, no. 17, pp. 2934–2942, 2004.

[21] H. de Jong, "Modeling and simulation of genetic regulatory systems: a literature review," *Journal of Computational Biology*, vol. 9, no. 1, pp. 67–103, 2004.

[22] S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," *Journal of Theoretical Biology*, vol. 22, no. 3, pp. 437–467, 1969.

[23] E. O. Voit, *Computational Analysis of Biochemical Systems: A Practical Guide for Biochemists and Molecular Biologists*, Cambridge University Press, Cambridge, UK, 2000.

[24] M. Vilela, I.-C. Chou, S. Vinga, A. T. R. Vasconcelos, E. O. Voit, and J. S. Almeida, "Parameter optimization in S-system models," *BMC Systems Biology*, vol. 2, no. 1, article 35, 2008.

[25] M. A. Savageau, "Power-law formalism: a canonical nonlinear approach to modelling and analysis," in *Proceedings of the World Congress of Nonlinear Analysts*, vol. 92, pp. 3323–3334, 1996.

[26] P. D'Haeseleer, S. Liang, and R. Somogyi, "Genetic network inference: from co-expression clustering to reverse engineering," *Bioinformatics*, vol. 16, no. 8, pp. 707–726, 2000.

[27] D. C. Weaver, C. T. Workman, and G. D. Stormo, "Modeling regulatory networks with weight matrices," in *Proceedings of the Pacific Symposium on Biocomputing (PSB '99)*, vol. 4, pp. 112–123, January 1999.

[28] E. van Someren, L. F. A. Wessels, and M. Reinders, "Linear modelling of genetic networks from experimental data," in *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*, vol. 8, pp. 355–366, 2000.

[29] E. Mjolsness, D. H. Sharp, and J. Reinitz, "A connectionist model of development," *Journal of Theoretical Biology*, vol. 152, no. 4, pp. 429–453, 1991.

[30] E. Mjolsness, T. Mann, R. Castaño, and B. Wold, "From co-expression to co-regulation: an approach to inferring transcriptional regulation among gene classes from large-scale expression data," in *Neural Information Processing Systems*, 1999.

[31] J. Vohradsky, "Neural model of the genetic network," *The Journal of Biological Chemistry*, vol. 276, no. 39, pp. 36168–36173, 2001.

[32] M. Wahde and J. Hertz, "Coarse-grained reverse engineering of genetic regulatory networks," *BioSystems*, vol. 55, no. 1–3, pp. 129–136, 2000.

[33] M. Wahde and J. Hertz, "Modeling genetic regulatory dynamics in neural development," *Journal of Computational Biology*, vol. 8, no. 4, pp. 429–442, 2001.

[34] R. Xu, D. C. Wunsch II, and 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, pp. 681–692, 2007.

[35] D. Marbach, C. Mattiussi, and D. Floreano, "Replaying the evolutionary tape: biomimetic reverse engineering of gene networks," *Annals of the New York Academy of Sciences*, vol. 1158, no. 1, pp. 234–245, 2009.

[36] P. D'Haeseleer, *Reconstructing gene networks from large-scale gene expression data [Ph.D. thesis]*, University of New Mexico, 2000.

[37] E. P. van Someren, L. F. A. Wessels, and M. J. T. Reindersm, "Genetic network models: a comparative study," in *Proceedings of the International Symposium on Biomedical Optics (BiOS '01)*, pp. 236–247, International Society for Optics and Photonics, San Jose, Calif, USA, 2001.

[38] D. C. Weaver, C. T. Workman, and G. D. Stormo, "Modeling regulatory networks with weight matrices," in *Pacific Symposium on Biocomputing*, vol. 4, pp. 112–123, 1999.

[39] Z. Bar-Joseph, G. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon, "A new approach to analyzing gene expression time series data," in *Proceedings of the 6th Annual International Conference on Computational Biology (RECOMB '02)*, pp. 39–48, ACM, April 2002.

[40] S. Kimura, K. Ide, A. Kashihara et al., "Inference of S-system models of genetic networks using a cooperative coevolutionary algorithm," *Bioinformatics*, vol. 21, no. 7, pp. 1154–1163, 2005.

[41] O. R. Gonzalez, C. Küper, K. Jung, P. C. Naval Jr., and E. Mendoza, "Parameter estimation using simulated annealing for S-system models of biochemical networks," *Bioinformatics*, vol. 23, no. 4, pp. 480–486, 2007.

[42] S. Kikuchi, D. Tominaga, M. Arita, K. Takahashi, and M. Tomita, "Dynamic modelling of genetic networks using a genetic algorithm and S-system," *Bioinformatics*, vol. 19, no. 5, pp. 643–650, 2003.

[43] S.-Y. Ho, C.-H. Hsieh, F.-C. Yu, and H.-L. Huang, "An intelligent two-stage evolutionary algorithm for dynamic pathway identification from gene expression profiles," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 4, no. 4, pp. 648–660, 2007.

[44] R. Xu, G. K. Venayagamoorthy, and D. C. Wunsch II, "Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization," *Neural Networks*, vol. 20, no. 8, pp. 917–927, 2007.

[45] H. W. Ressom, Y. Zhang, J. Xuan, Y. Wang, and R. Clarke, "Inference of gene regulatory networks from time course gene expression data using neural networks and swarm intelligence," in *Proceedings of the IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology (CIBCB '06)*, pp. 1–8, IEEE, Toronto, Canada, 2006.

[46] K. Kentzoglanakis and M. Poole, "A swarm intelligence framework for reconstructing gene networks: searching for biologically plausible architectures," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 2, pp. 358–371, 2012.

[47] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[48] Y. Maki, T. Ueda, M. Okamoto et al., "Inference of genetic network using the expression profile time course data of mouse P19 cells," *Genome Informatics*, vol. 13, pp. 382–383, 2002.

[49] N. Noman and H. Iba, "Inferring gene regulatory networks using differential evolution with local search heuristics," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 4, no. 4, pp. 634–647, 2007.

[50] Z. Bar-Joseph, G. K. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon, "Continuous representations of time-series gene expression data," *Journal of Computational Biology*, vol. 10, no. 3-4, pp. 341–356, 2003.

[51] M. S. Dasika, A. Gupta, C. D. Maranas, and J. D. Varner, "A mixed integer linear programming (MILP) framework for inferring time delay in gene regulatory networks," in *Pacific Symposium on Biocomputing*, vol. 9, pp. 474–485, 2003.

[52] E. P. van Someren, L. F. A. Wessels, M. J. T. Reinders, and E. Backer, "Robust genetic network modeling by adding noisy data," in *Proceedings of the IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing*, 2001.

[53] C. Spieth, F. Streichert, N. Speer, and A. Zell, "Optimizing topology and parameters of gene regulatory network models from time-series experiments," in *Genetic and Evolutionary Computation—GECCO 2004*, pp. 461–470, Springer, Berlin, Germany, 2004.

[54] E. Keedwell and A. Narayanan, "Discovering gene networks with a neural-genetic hybrid," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 2, no. 3, pp. 231–242, 2005.

[55] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, vol. 1, pp. 39–43, Nagoya, Japan, October 1995.

[56] A. Banks, J. Vincent, and C. Anyakoha, "A review of particle swarm optimization. Part I: background and development," *Natural Computing*, vol. 6, no. 4, pp. 467–484, 2007.

[57] A. Banks, J. Vincent, and C. Anyakoha, "A review of particle swarm optimization. Part II: hybridisation, combinatorial, multi-criteria and constrained optimization, and indicative application," *Natural Computing*, vol. 7, no. 1, pp. 109–124, 2008.

[58] E. Elbeltagi, T. Hegazy, and D. Grierson, "Comparison among five evolutionary-based optimization algorithms," *Advanced Engineering Informatics*, vol. 19, no. 1, pp. 43–53, 2005.

[59] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284 of *Studies in Computational Intelligence*, pp. 65–74, Springer, Berlin, Germany, 2010.

[60] A. Alihodzic and M. Tuba, "Improved bat algorithm applied to multilevel image thresholding," *The Scientific World Journal*, vol. 2014, Article ID 176718, 16 pages, 2014.

[61] N. Noman and H. Iba, "Reverse engineering genetic networks using evolutionary computation," *Genome Informatics Series*, vol. 16, no. 2, pp. 205–214, 2005.

[62] L. Palafox, N. Noman, and H. Iba, "Reverse engineering of gene regulatory networks using dissipative particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 4, pp. 577–587, 2013.

[63] T. Schaffter, D. Marbach, and D. Floreano, "GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods," *Bioinformatics*, vol. 27, no. 16, pp. 2263–2270, 2011.

[64] M. Ronen, R. Rosenberg, B. I. Shraiman, and U. Alon, "Assigning numbers to the arrows: parameterizing a gene regulation network by using accurate expression kinetics," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 16, pp. 10555–10560, 2002.

[65] A. Greenfield, A. Madar, H. Ostrer, and R. Bonneau, "DREAM4: combining genetic and dynamic information to identify biological networks and dynamical models," *PLoS ONE*, vol. 5, no. 10, Article ID e13397, 2010.