

Reverse Engineering Genetic Networks Using Evolutionary Computation

Nasimul Noman

Hitoshi Iba

noman@iba.k.u-tokyo.ac.jp

iba@iba.k.u-tokyo.ac.jp

Department of Frontier Informatics, Graduate School of Frontier Sciences, The University of Tokyo, 5-1-5 Kashiwanoha, Kashiwa-shi, Chiba 277-8561, Japan

Abstract

This paper proposes an improved evolutionary method for constructing the underlying network structure and inferring effective kinetic parameters from the time series data of gene expression using decoupled S-system formalism. We employed Trigonometric Differential Evolution (TDE) as the optimization engine of our algorithm for capturing the dynamics in gene expression data. A more effective fitness function for attaining the sparse structure, which is the hallmark of biological networks, has been applied. Experiments on artificial genetic network show the power of the algorithm in constructing the network structure and predicting the regulatory parameters. The method is used to evaluate interactions between genes in the SOS signaling pathway in *Escherichia coli* using gene expression data.

Keywords: S-system, gene regulatory network, DNA microarray, differential evolution, SOS DNA repair network, *E. coli*

1 Introduction

The recent development of microarray, that allows us to monitor the transcriptomes on a genome-wide scale, has grown interest among many researchers to use the model-based identification methods for inferring the possible regulatory architectures in genetic networks. To represent the regulatory relations among genes several genetic network models have been proposed [1, 3, 10, 14]. The modeling spectrum ranges from abstract Boolean descriptions to detailed Differential Equation based models, where every representation has its advantages and limitations. Given a dynamic model of gene interactions, the problem of gene network inference is equivalent to learning the structural and functional parameters from the time series representing the gene expression kinetics, i.e. the network architecture is reverse engineered from its activity profiles [17].

Among the familiar models for describing biochemical networks, a well studied one is S-system which is rich enough to reasonably capture the nonlinearity of genetic regulation [18]. Savageu has proposed S-system based on power law formalism described as follows

$$\frac{dX_i}{dt} = \alpha_i \prod_{j=1}^N X_j^{g_{ij}} - \beta_i \prod_{j=1}^N X_j^{h_{ij}}, \quad (1)$$

where N is the number of network components or reactants (X_i). The terms g_{ij} and h_{ij} represent interactive affectivity of X_j to X_i . From the biological point of view, the two terms in right-hand side of (1) represent the productive and inhibitory regulation respectively, influencing the variable at the left hand side of the equation. The parameters that define the S-system are: $\{\alpha, \beta, g, h\}$. In the biochemical engineering context, the non-negative parameters α_i , β_i are called *rate constants*, and real-valued exponents g_{ij} and h_{ij} are referred to as *kinetic orders* [7].

Tominaga *et al.* [23] formulated the S-system based gene network estimation as an optimization problem and they used *Genetic Algorithm* (GA) to estimate model parameters. Since methods for finding analytic solution for this problem is almost impracticable, use of *Evolutionary Computation* (EC) has become more feasible and popular method among researchers [2, 8, 12, 19]. In the form of optimization problem each set of parameters estimated for the S-system model of a genetic network is evaluated as follows. Suppose that $X_{i,cal,t}$ is gene expression level of gene X_i at time t calculated numerically by solving the system of differential equation of (1) for the estimated parameter set, and $X_{i,exp,t}$ represents the experimentally observed gene expression level of X_i at time t . Sum of the relative squared error between $X_{i,cal,t}$ and $X_{i,exp,t}$ is taken as the relative standard error f for fitness estimation [23]

$$f = \sum_{i=1}^N \sum_{t=1}^T \left\{ \left(\frac{X_{i,cal,t} - X_{i,exp,t}}{X_{i,exp,t}} \right)^2 \right\}, \quad (2)$$

where N is the number of state variables, T is the number of sampling points of the experimental data. The problem is to find a set of parameters that minimizes f .

The problem has the difficulty of high-dimensionality, since $2N(N+1)$ S-system parameters must be determined in order to solve the set of differential equations (1). And estimation of parameters for a $2N(N+1)$ dimensional function optimization problem often causes bottlenecks and fitting the model to experimentally observed responses (time course of relative state variables or reactants) is never straightforward and is almost always difficult. Therefore the application of S-system model has been limited to inference of small-scale gene networks only.

In order to deal with problem of high-dimensionality a decoupling of the original model has been performed [9, 11]. This decoupled S-system model has been proven to be very efficient and effective for inferring larger gene networks. In this paper we propose an improved algorithm that finds the parameter values of decomposed S-system model based networks with higher accuracy using *Trigonometric Differential Evolution* (TDE). We used a modified fitness function and an enhanced algorithm for estimating the correct network topology and parameter values. Numerical experiments show that the proposed enhancements attain the network structure and parameter values very accurately and efficiently. We also tested our algorithm against the experimental data related to the SOS DNA repair network of the *Escherichia coli* bacterium. The paper is organized as follows. In the next section we present the S-system model for genetic networks in its disassociated form and the proposed fitness function for evaluating candidate networks. In Section 3 our proposed algorithm for parameter estimation of S-system model based gene networks is presented. Section 4 reports the experiments and the results to verify the effectiveness of the proposed method. Finally Section 5 concludes the paper.

2 Decomposition Strategy

2.1 Decoupled S-system Model

As mentioned earlier, for resolving the problem of high-dimensionality a problem decomposition strategy has been suggested, where the original optimization problem is divided into N subproblems [9, 11]. In each of these subproblems the parameter values of gene i is estimated for realizing the temporal profile of gene expression. In other words, this disassociation technique divides a $2N(N+1)$ dimensional optimization problem into N subproblems of $2(N+1)$ dimension. In i -th subproblem for gene i $X_{i,cal,t}$ is calculated by solving the following differential equation instead

$$\frac{dX_i}{dt} = \alpha_i \prod_{j=1}^N Y_j^{g_{ij}} - \beta_i \prod_{j=1}^N Y_j^{h_{ij}}, \quad (3)$$

where

$$Y_j = \begin{cases} X_j, & \text{if } j = i \\ \hat{X}_j, & \text{otherwise} \end{cases} \quad (4)$$

\hat{X}_j is an estimation of gene expression level obtained not by solving any differential equation, but by making direct estimation from the observed time-series data. In this work we have applied linear spline interpolation [15] for such estimation.

And sum of squared relative error between experimental and calculated gene expression levels of gene i is used as a means of evaluating the fitness in subproblem i . So the objective function of the subproblem corresponding to the i -th gene becomes

$$f_i = \sum_{t=1}^T \left\{ \left(\frac{X_{i,cal,t} - X_{i,exp,t}}{X_{i,exp,t}} \right)^2 \right\}. \quad (5)$$

And in subproblem i we try to estimate the parameters $\{\alpha_i, \beta_i, g_{ij}, h_{ij} (j = 1, \dots, N)\}$ for gene i that minimizes f_i .

2.2 Fitness Function

Continued development in the community has led to the discovery of one major pitfall for S-system based gene regulatory network estimation is identifying the sparse structure of the biological networks. Because of the high degree-of-freedom of the model and the deceptive nature of the problem, solutions often converge to different local minima each of which reproduces almost same time-course. So any method attempting to reproduce the time dynamics only, fail to obtain the skeletal structure [8]. Use of an additional term called *pruning term* or *penalty term* for augmentation of the fitness equation was very successful to deal with this difficulty [8, 9, 13]. For identifying the sparse network structure which is more usual for real biological systems the basic fitness function of (2) was extended by adding a term based on Laplacian regularization term using all *kinetic orders* in [8, 13]. Kimura *et al.* added another more effective *penalty term* to the objective function of (5), as follows [9].

$$f_i = \sum_{t=1}^T \left(\frac{X_{i,cal,t} - X_{i,exp,t}}{X_{i,exp,t}} \right)^2 + c \sum_{j=1}^{N-I} (|G_{ij}| + |H_{ij}|), \quad (6)$$

where G_{ij} and H_{ij} are given by rearranging g_{ij} and h_{ij} , respectively, in ascending order of their absolute values. (i.e., $|G_{i1}| \leq |G_{i2}| \leq \dots \leq |G_{iN}|$ and $|H_{i1}| \leq |H_{i2}| \leq \dots \leq |H_{iN}|$). And I is the maximum cardinality of the network and c is the penalty constant. This *pruning term* will penalize only when the number of genes that directly affect the i -th gene is higher than the maximum indegree I , causing most of the genes to disconnect when this penalty term is applied. However very few genes affect both the synthesis process and degradation process of a specific genes. So we modified the fitness of (6) as follows:

$$f_i = \sum_{t=1}^T \left(\frac{X_{i,cal,t} - X_{i,exp,t}}{X_{i,exp,t}} \right)^2 + c \sum_{j=1}^{N-I} (|K_{ij}|), \quad (7)$$

where K_{ij} are the *kinetic orders* of gene i sorted in ascending order of their absolute values. Using (7) instead of (6) we could reduce the number of false predicted regulations. In other words we can identify the zero valued parameters increasingly and thus obtain the skeletal network structure more precisely.

3 Method

3.1 Trigonometric Differential Evolution

As mentioned earlier, EC has become very popular approach for gene network inference problem. In this section we give a brief review of Trigonometric Differential Evolution (TDE) algorithm that we applied as the optimization engine in our algorithm described in next section. TDE is an extension of Differential Evolution (DE) an effective, efficient and robust optimization method [21] capable of handling nondifferentiable, nonlinear and multimodal objective functions [20, 21].

In DE new individuals are generated by the combination of randomly chosen individuals from the population. Specifically, for each individual x_G^i , $i = 1, \dots, P$, where G denotes the current generation, a new individual y_{G+1}^i is generated according to the following equation

$$y_{G+1}^i = x_G^j + F(x_G^k - x_G^l), \quad (8)$$

where j, k and l are random integers such that j, k and $l \in \{1, \dots, P\}$ and $i \neq j \neq k \neq l$ and F is called *scaling factor* or *amplification factor*. This operation is similar to what is commonly known as *mutation* to EC community. In order to achieve higher diversity the mutated individual y_{G+1}^i is mated with the current population member x_G^i using a *crossover* operation to generate the *offspring* or *trial individual* x_{G+1}^i . The genes of x_{G+1}^i are randomly inherited from x_G^i or y_{G+1}^i determined by a parameter called *crossover factor* CF , i.e. if $r \leq CF$ (where r is a uniform random number in $[0, 1]$) then it is inherited from x_G^i otherwise from y_{G+1}^i . Finally the offspring is evaluated and replaces its parent x_G^i in next generation if and only if its fitness is better than that of its parent. This is the *selection* process.

Recently Fan and Lampinen have proposed a Trigonometric Mutation Operation (TMO) for DE to accelerate its convergence rate and robustness [5] which is defined as

$$y_{G+1}^i = (x_G^j + x_G^k + x_G^l)/3 + (p_k - p_j)(x_G^j - x_G^k) + (p_l - p_k)(x_G^k - x_G^l) + (p_j - p_l)(x_G^l - x_G^j), \quad (9)$$

where

$$p_j = |f(x_G^j)|/p', \quad p_k = |f(x_G^k)|/p', \quad p_l = |f(x_G^l)|/p' \quad \text{and} \quad p' = |f(x_G^j)| + |f(x_G^k)| + |f(x_G^l)|.$$

This TMO is applied with a probability of M_t and the regular mutation operation given by (8) is applied with probability $(1 - M_t)$ and this modified DE algorithm is called Trigonometric mutation DE (TDE). Since the trigonometric mutation operation is a rather greedy search operator, this modification of the DE algorithm makes it possible to straightforwardly adjust the balance between the convergence rate and the robustness through the newly introduced parameter, M_t . The greediness of the algorithm can be tuned conveniently by increasing or decreasing M_t . Experimental results have shown that TDE has good convergence properties, outperforms other well known EAs [5] and effective in genetic network inference [13]. Because of these admirable properties we have chosen TDE as optimization tool in our algorithm for gene network inference problem which is explained in next section.

3.2 Proposed Algorithm

In our algorithm for estimating the S-system parameters, we optimized the objective function (7) for each subproblem i ($i = 1, \dots, N$) using the TDE algorithm with a local search procedure explained later. Here we explain the optimization process for a specific subproblem. Same is applied for other subproblems to obtain a complete set of parameters for the full network.

In order to estimate the parameters more accurately and to avoid premature convergence to some local minima we performed optimization using a two step method. In each phase the parameter values of the gene are represented as an individual of TDE. In the first stage we perform many trials of optimization starting with different random initial individuals. In each of these trials, at the end

of the optimization using TDE augmented with local search procedure we obtain a solution for the subproblem i.e. a set of parameters for the target gene. However the optimization process possibly converged to a local optimum and may have failed to attain actual parameter set. And because of local convergence it may lose some essential regulatory interaction among the genes. In other words we can say, due to convergence to local minima some parameter value could go down to zero, which is not actually zero in the target parameter set. Since trials are repeated with different initial values, candidate solutions are obtained as possible different local minima. To obtain a more robust and accurate solution we perform double optimization using elite individuals from different trials in second step of our algorithm. Double optimization could automatically detect the essential parameters by optimizing multiple local minima once again [8]. The solutions obtained from the multiple local minima of the first stage retained some essential parameters. So applying another optimization on these solutions we can identify the correct regulations, accurate strength of the regulation and avoid deletion of any necessary parameter.

In each phase of optimization the overall procedure of estimating the model parameters for each subproblem is as follows:

1. Prepare initial population P_G with candidate solutions
2. Create the new generation P_{G+1} of candidate solution applying recombination/selection operation of TDE
3. Apply local search to the best individual and a randomly selected individual of the new generation P_{G+1}
4. Stop if the termination criteria satisfied. Otherwise Set $G = G + 1$ and go to Step 2

In phase 1 the initial population P_G is created randomly. And Γ trials of this phase 1 is repeated. In Phase 2 the elite individuals from different trials of Phase 1 together with some random individuals are used for initialization.

3.3 Local Search Procedure

In order to provide an effective global optimization method some metaheuristics or local searches are often embedded inside the evolutionary algorithm. Genetic Algorithms (GAs) hybridized with local refinement procedures are known as Memetic Algorithm (MAs) which are motivated to take advantage of both the exploration abilities of GA and the exploitation abilities of local search. Therefore we introduce a local search method inside the TDE algorithm. In our local refinement procedure we perform a greedy search operation around the best individual and a random individual of each generation. The local search around the best individual and a random individual will accelerate the optimization process as well as maintain the diversity of the population. The local searching is performed as follows: All the *kinetic orders* are sorted in ascending order of their absolute values. Then the *kinetic order* of the lowest absolute value is set to zero. And the fitness of this new individual is evaluated. If this modification improves the fitness of the individual then new solution is accepted otherwise it's parent solution is restored. And this process repeated for all the kinetic order in increasing order of their absolute values. This local search process allows us to identify the zero valued parameters by mutating them in the increasing order of their strength and thus helps us to identify the skeletal network structure. And the restore capability of the greedy search also allows to recover from wrong elimination of any essential regulation. Hybridizing this greedy local search procedure with the TDE algorithm we can identify the sparse network structure efficiently and the strength of regulations more accurately.

4 Experiment

4.1 Experiment 1: Noise Free Time Series

To confirm the effectiveness of the proposed algorithm we experimented with an artificial genetic network inference problem. As the target network we used a small-scale S-system model with the parameter set listed in Table 1 [8]. This network was first studied by Tominaga in [23] and later many others have experimented with it [8, 9, 13, 22]. Hence this network, which represents a typical gene interaction system consisting of 5 genes, has become like a benchmark network for evaluating the performance of optimization algorithms for S-system models.

If an insufficient amount of time series data is used for estimating the parameters for S-system model many candidate solutions will evolve due to the high-degree of freedom of the model. This is because, it is only one path in a phase diagram and from such a single path no general conclusions about the overall behavior of the dynamic system can be drawn [22]. Therefore Kikuchi *et al.* [8] and Kimura *et al.* [9] had used 50 and 75 time series data, respectively, for solving this 5 gene network [8, 9]. We have also used the same sets of 50 time series data from [8]. The sets of time-series were obtained by solving the set of differential equations (1) on the model in Table 1. In our experiment we used 11 sampling points from each time-series of gene expression.

Table 1: S-system parameters for target network

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	5.0	0.0	0.0	1.0	0.0	-1.0	10.0	2.0	0.0	0.0	0.0	0.0
2	10.0	2.0	0.0	0.0	0.0	0.0	10.0	0.0	2.0	0.0	0.0	0.0
3	10.0	0.0	-1.0	0.0	0.0	0.0	10.0	0.0	-1.0	2.0	0.0	0.0
4	8.0	0.0	0.0	2.0	0.0	-1.0	10.0	0.0	0.0	0.0	2.0	0.0
5	10.0	0.0	0.0	0.0	2.0	0.0	10.0	0.0	0.0	0.0	0.0	2.0

4.1.1 Experimental Setup

The conditions of our experiments were as follows. The search regions of the parameters were $[0.0, 15.0]$ for α_i and β_i , and $[-3.0, 3.0]$ for g_{ij} and h_{ij} . The maximum cardinality I was chosen 5, and the penalty coefficient c was 1.0. The parameter values for TDE algorithm were $F = 0.5$, $CF = 0.8$ and $M_t = 0.05$, population size was 60 and the maximum number of generation in each trial of Phase 1 and in Phase 2 was 850. In Phase 1 we evolved 5 ($\Gamma = 1, \dots, 5$) independent trial solutions from which we selected elite individuals for optimization in Phase 2. Our algorithm was implemented in Java language and the time required for solving each subproblem was approximately 10 minutes using a PC with 1700 MHz Intel Pentium processor and 512 MB of RAM.

In order to reduce the computational burden a structure skeletalizing was applied in a similar fashion used by Tominaga *et al.* [23]. If the absolute value of a parameter is less than a threshold value δ then structure skeletalizing reset it to zero. This process reduces the computational cost as well as helps to identify the zero valued parameters. In our experiment $\delta = 0.001$ was used. We used 5 repetitions for each experiment to assure soundness of our stochastic search algorithm.

4.1.2 Result

Table 2 shows the parameters estimated by our algorithm in a typical run (not the best out of 5 runs). As shown in the table our model was able to attain the over all network structure and parameter values were also very close to targeted values. Many of the zero valued parameters were identified correctly and the others are close enough to zero to indicate false positive interactions.

Table 2: Parameters estimated using 10 sets noise-free time series

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	4.762	-0.021	-0.021	0.993	0.0	-1.013	9.607	1.916	0.0	0.0	0.0	0.0
2	10.080	1.990	-0.001	0.035	0.0	0.0	9.817	0.0	1.938	0.0	0.012	0.0
3	9.823	0.0	-1.000	-0.008	0.0	-0.001	9.835	0.0	-1.000	2.031	0.0	0.0
4	7.182	0.0	-0.036	2.039	-0.052	-1.044	9.415	0.0	0.0	0.0	2.034	0.0
5	10.103	0.0	0.005	0.050	1.997	-0.003	10.049	0.0	0.0	0.0	0.0	2.005

4.2 Experiment 2: Noisy Time Series

Noise is inevitable in the microarray data and the real challenge lies in designing inference algorithm capable of constructing the network from noisy data. In this section we test the performance of our proposed algorithm simulating a noisy real world environment i.e. conducting the experiments with the set of noisy time series data.

4.2.1 Experimental Setup

As the target model we select the same network used in Experiment 1 with the same target parameter set. Data points were generated using the same sets of initial expression used in Experiment 1. We added 5% Gaussian noise to the time-series data in order to simulate the measurement error between true expression and observed expression. 11 sample points from each time course were used for optimization. The rest of the settings were same as in the previous experiments.

4.2.2 Results

The set of parameter inferred by our method in a typical run using the noisy data is presented in Table 3. Even using noisy data the proposed method were successful in identifying all the correct regulations and a few false regulations. The number of false-negative interactions was zero. In some cases the estimation of *kinetic constants* were pretty far away from the target value and some false predicted parameter values were too large to ignore. Though all the parameter values were not estimated accurately but the underlying regulations were always identified correctly. Such indication may turn out to be very useful for biologist to design additional experiments or to develop conjectures. However use of additional time courses can decrease the number of false predictions as well as more accurate estimation of parameter values.

Table 3: Parameters estimated using 10 sets of noisy time series

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	5.001	-0.063	0.037	1.137	0.0	-1.017	9.906	2.236	0.0	0.0	0.0	0.0
2	9.949	2.114	-0.052	0.0	0.020	0.0	15.000	0.0	2.068	0.0	0.0	0.285
3	7.210	0.0	-1.083	-0.339	0.121	0.0	7.740	0.0	-1.037	3.0	0.0	0.0
4	5.533	0.0	0.0	2.449	-0.307	-1.103	8.434	0.0	0.0	0.0	2.971	0.630
5	10.356	0.059	0.0	-0.148	1.873	0.020	8.289	0.0	0.0	0.0	0.0	1.633

4.3 Experiment 3: Analysis of Microarray Data

We tested the proposed algorithm on the well-known SOS DNA repair network in *Escherichia coli*. Exposure of *E. coli* to agents or conditions that damage DNA induces this gene network for repairing of DNA. The entire system involves more than 100 genes [6]. The expression of the genes in the SOS regulatory network is controlled by a complex circuitry involving the *RecA* and *LexA* proteins. In an un-induced cell, the product of the *lexA* gene acts as the repressor of more than 20 genes, including the *recA* and *lexA* genes, by binding to the promoter region of these genes. One of the SOS proteins,

RecA, acts as a sensor of DNA damage: by binding to single-stranded DNA, becomes activated and mediates *LexA* destruction. The drop in *LexA* levels activates the SOS genes. When damage has been repaired or bypassed, the level of activated *RecA* drops, *LexA* accumulates and represses the SOS genes, and cell returns to its initial state [14, 16].

4.3.1 Experimental Data Set

We downloaded experimental data from the homepage of Uri Alon Lab [24]. Data are expression kinetics of 8 genes (*uvrD*, *lexA*, *umuD*, *recA*, *uvrA*, *uvrY*, *ruvA* and *polB*) of the SOS DNA repair network. The measurement technology is based on the property of GFPs (green fluorescent proteins). Alon *et al.* have developed a system for obtaining very precise kinetics [16]. Measurements are done after irradiation of the DNA at the initial time with UV light. Four experiments are done for various light intensities (Exp. 1&2: $5 Jm^{-2}$, Exp. 3&4: $20 Jm^{-2}$). Each experiment composed of 50 instants evenly spaced by 6 minutes intervals and 8 genes are monitored.

In our experiment we used all the data from Alon's experiment 1 and 3, i.e. we used 50×2 sampling points for each gene. The data were normalized with in the range (0, 1] and all the zero expression levels were replaced with a very small value. 5 runs were carried out to assure the statistical significance of the probabilistic search.

4.3.2 Results

As microarray data are noisy the results were much dispersed so we applied Z-score [4] to analyze which regulations are more significant and less diverse than others. The analysis of Z-score is more qualitative but sufficient for comparing with the existing knowledge or suggesting new regulation. We considered only those regulations which have Z-score value above the threshold $Z_{th} = 1.5$, which was set empirically.

Table 4: Predicted regulations for SOS repair network

Gene	Predicted Regulators
<i>uvrD</i>	<i>uvrD</i> , <i>uvrA</i> , <i>uvrY</i> , <i>polB</i>
<i>lexA</i>	<i>lexA</i> , <i>recA</i> , <i>umuD</i>
<i>umuD</i>	<i>lexA</i>
<i>recA</i>	<i>lexA</i>
<i>uvrA</i>	<i>lexA</i> , <i>recA</i> , <i>uvrA</i>
<i>uvrY</i>	<i>recA</i> , <i>polB</i>
<i>ruvA</i>	
<i>polB</i>	<i>polB</i>

Table 4 shows the regulators of different genes in SOS repair network identified by our algorithm. Looking at the table it can be found that the *lexA* regulation of *lexA*, *umuD*, *recA* and *uvrA* has been correctly identified. The activation of *lexA* by *recA* was also correctly predicted. An unknown regulation of *uvrA* by *recA* was identified. Perrin *et al.* also identified this regulation in their experiment and hypothesized that this could correspond to the indirect regulation $recA \rightarrow RecA \dashv LexA \dashv uvrA$ [14]. Regulation of *lexA* by *umuDC* is also known. Some other regulations were inferred, those are either novel regulatory pathways or false-positive findings. Moreover in the microarray experiment the UV light shock was not sufficient to lead to the functioning of all SOS genes. From the expression profile it can be found that several genes were not induced sufficiently during the experiment. These genes are activated only when the damage is sufficiently high and their activation would have been useful to identify interrelation among the genes.

5 Conclusion

In this paper we have proposed a more effective evolutionary approach for inferring gene regulatory network using decoupled S-system formalism. Grounded on the fact that not too many genes regulate both the synthesis and degradation process of other genes we modified the fitness function proposed by Kimura *et al.* for reducing the number of false positive predictions. For obtaining a more robust set of parameter values we applied double optimization using Trigonometric Differential Evolution algorithm. For identifying the sparse network structure, which is more common in biological system, we also embedded a hill-climbing local search procedure with in the general framework of our evolutionary algorithm. Numerical experiments using a well-studied small scale artificial network both in noise-free and noisy environment showed the effectiveness of the method in identifying the network structure and parameter values. We also perform some analysis of microarray data for reconstructing the SOS DNA repair network of *E. coli*. The proposed algorithm identified the regulations of gene *lexA* and some other known regulations correctly.

The proposed method works well for small networks, or medium scale networks. But large scale genetic networks consisting of hundreds or thousands of genes are still out of the scope of the method in the current form. For reconstruction of large scale convoluted networks of complex organisms we need improvement in methodology, algorithmic efficiency, computing power, technical accuracy and additional domain knowledge. However, we believe the speed up achieved by the proposed method in predicting S-system model parameters will make it useful for larger network inference.

References

- [1] Akutsu, T., Miyano, S., and Kuhara, S., Identification of genetic networks from a small number of gene expression patterns under the boolean network model, *Pac. Symp. Biocomput.*, 4:17–28, 1999.
- [2] Ando, S., Sakamoto, E., and Iba, H., Evolutionary modeling and inference of gene network, *Information Sciences*, 145(3-4):237–259, 2002.
- [3] Bower, J.M. and Bolouri, H., *Computational Modeling of Genetic and Biochemical Networks*, The MIT Press, 2001.
- [4] D’Haeseller, P., Liang, S., and Somogyi, R., Genetic network inference: From co-expression clustering to reverse engineering, *Bioinformatics*, 16(8):707–726, 2000.
- [5] Fan, H.-Y. and Lampinen, J., A trigonometric mutation operation to differential evolution, *J. Global Optim.*, 27(1):105–129, 2003.
- [6] Gardner, T.S., Bernardo, D.di, Lorenz, D., and Collins, J.J., Inferring genetic networks and identifying compound mode of action via expression profiling, *Science*, 301(5629):102–105, 2003.
- [7] Irvine, D. and Savageau, M., Efficient solution of nonlinear ordinary differential equations expressed in S-system canonical form, *SIAM J. Numerical Analysis*, 27(3):704–735, 1990.
- [8] Kikuchi, S., Tominaga, D., Arita, M., Takahashi, K., and Tomita, M., Dynamic modeling of genetic networks using genetic algorithm and S-system, *Bioinformatics*, 19(5):643–650, 2003.
- [9] Kimura, S., Ide, K., Kashihara, A., Kano, M., Hatakeyama, M., Masui, R., Nakagawa, N., Yokoyama, S., Kuramitsu, S., and Konagaya, A., Inference of S-system models of genetic networks using cooperative coevolutionary algorithm, *Bioinformatics*, 21(7):1154–1163, 2005.
- [10] Maki, Y., Tominaga, D., Okamoto, M., Watanabe, S., and Eguchi, Y., Development of a system for the inference of large scale genetic networks, *Pac. Symp. Biocomput.*, 6:446–458, 2001.

- [11] Maki, Y., Ueda, T., Okamoto, M., Uematsu, N., Inamura, K., Uchida, K., Takahashi, Y., and Eguchi, Y., Inference of genetic network using the expression profile time course data of mouse p19 cells, *Genome Informatics*, 13:382–383, 2002.
- [12] Morishita, R., Imade, H., Ono, I., Ono, N., and Okamoto, M., Finding multiple solutions based on an evolutionary algorithm for inference of genetic networks by S-system, *Proc. Congress on Evolutionary Computations*, 615–622, 2003.
- [13] Noman, N. and Iba, H., Inference of gene regulatory networks using S-system and differential evolution, *Proc. Genetic and Evolutionary Computation Conference*, 439–446, 2005.
- [14] Perrin, B.-E., Ralaivola, L., Mazurie, A., Bottani, S., Mallet, J., and d’Alché-Buc, F., Gene networks inference using dynamic Bayesian networks, *Bioinformatics*, 19(Suppl 2):ii138–ii148, 2003.
- [15] Press, W., Teukolsky, S., Vetterling, W., and Flannery, B., *Numerical Recipes in C*, second edition, Cambridge University Press, 1995.
- [16] Ronen, M., Rosenberg, R., Shraiman, B.I., and Alon, U., Assigning numbers to the arrows: Parameterizing a gene regulation network by using accurate expression kinetics, *Proc. Natl. Acad. Sci. USA*, 99:10555–10560, 2002.
- [17] Sakamoto, E., and Iba, H., Inferring a system of differential equations for a gene regulatory network by using genetic programming, *Proc. Congress on Evolutionary Computation*, 720–726, 2001.
- [18] Savageau, M.A., 20 years of S-systems, in Voit, E. (ed.), *Canonical Nonlinear Modeling. S-systems Approach to Understand Complexity*, 1–44. Van Nostrand Reinhold, 1991.
- [19] Speith, C., Streichert, F., Speer, N., and Zell, A., Optimizing topology and parameters of gene regulatory network models from time-series experiments, *Proc. Genetic and Evolutionary Computation Conference*, 461–470, 2004.
- [20] Storn, R., System design by constraint adaptation and differential evolution, *IEEE Trans. on Evolutionary Computation*, 3(1):22–34, 1999.
- [21] Storn, R. and Price, K., Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.*, 11(4):341–359, 1997.
- [22] Streichert, F., Planatscher, H., Spieth, C., Ulmer, H., and Zell, A., Comparing genetic programming and evolution strategies on inferring gene regulatory networks, *Proc. Genetic and Evolutionary Computation Conference*, 471–480, 2004.
- [23] Tominaga, D., Koga, N., and Okamoto, M., Efficient numerical optimization algorithm based on genetic algorithm for inverse problem, *Proc. Genetic and Evolutionary Computation Conference*, 251–258, 2000.
- [24] <http://www.weizmann.ac.il/mcb/UriAlon/>.