An Intelligent Two-Stage Evolutionary Algorithm for Dynamic Pathway Identification from Gene Expression Profiles

Shinn-Ying Ho, Chih-Hung Hsieh, Fu-Chieh Yu, and Hui-Ling Huang

Abstract—From gene expression profiles, it is desirable to rebuild cellular dynamic regulation networks to discover more delicate and substantial functions in molecular biology, biochemistry, bioengineering, and pharmaceutics. The S-system model is suitable to characterize biochemical network systems and capable of analyzing the regulatory system dynamics. However, the inference of an S-system model of $N\text{-gene}$ genetic networks has $2N(N+1)$ parameters in a set of nonlinear differential equations to be optimized. This paper proposes an intelligent two-stage evolutionary algorithm (iTEA) to efficiently infer the S-system models of genetic networks from time-series data of gene expression. To cope with the curse of dimensionality, the proposed algorithm consists of two stages, where each uses a divide-and-conquer strategy. The optimization problem is first decomposed into N subproblems having $2(N+1)$ parameters each. At the first stage, each subproblem is solved using a novel intelligent genetic algorithm (IGA) with intelligent crossover based on an orthogonal experimental design (OED). At the second stage, the obtained N solutions to the N subproblems are combined and refined using an OED-based simulated annealing algorithm for handling noisy gene expression profiles. The effectiveness of iTEA is evaluated using simulated expression patterns with and without noise running on a single-processor PC. It is shown that 1) IGA is efficient enough to solve subproblems, 2) IGA is significantly superior to the existing method GA with simplex crossover (SPXGA), and 3) iTEA performs well in inferring S-system models for dynamic pathway identification.

Index Terms—Divide and conquer, evolutionary algorithm, genetic network, orthogonal experimental design, pathway identification, S-system model.

1 Introduction

Traditional biological experiments mainly concentrate on small-scale or local reactions among parts of complex biological system behavior. When faced with large-scale networks of enormous amounts of genes, proteins, and other metabolites, we cannot get rid of getting aid from other efficient tools. With widespread genomic research using microarray techniques, we can monitor global gene expression from genomic DNA one at a time [1]. Through microarray techniques, we can rebuild the cellular dynamic regulation networks from gene expression profiles that may imply the many functions of genes. The goal of constructing genetic network models is to reveal the regulation rules behind the gene expression profiles.

The numerous mathematical algorithms and models proposed to describe biochemical networks include [2]: the Boolean network model [3], [4], [5], the Bayesian network [6], [7], [8], [9], and the differential model or S-system model [10], [11], [12], [13], 14], [15], [16], [17], [18], [19], [20]. In Boolean network models, gene expression levels can refer to two situations: true or false. The Bayesian network model is able to deal with linear, nonlinear, and combinatorial problems and is also used to infer genetic networks. However, similarly to Boolean networks, it suffers from the same dilemma and is only applicable to acyclic structures [2], [6]. To cope with cyclic networks, some authors adopted the adapted dynamic Bayesian network [7], [9]. Another frequently used approach is to use differential equation models for the analysis of gene expression. The most popular model is referred to as the S-system model, which has been considered suitable to characterize biochemical network systems and capable to analyze the regulatory system dynamics. The model is a set of nonlinear differential equations of the following form [10], [11], [12], [13]:

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

where X_i represents the expression level of gene i and N is the number of genes in a genetic network. α_i and β_i are rate constants that indicate the direction of mass flow and must be positive. g_{ij} and h_{ij} are kinetic orders that reflect the intensity of interaction from gene j to i. For inferring an S-system model, it is necessary to estimate all of the $2N(N+1)$ S-system parameters (α_i, β_i, g_{ij}, and h_{ij}) from the experimental time-series data of gene expression.

Essentially, this reverse engineering problem is a large-scale parameter optimization problem that is time-consuming and intractable. Many researchers proposed various

- *S.-Y. Ho, C.-H. Hsieh, and F.-C. Yu are with the Institute of Bioinformatics, National Chiao Tung University, Hsinchu, 300 Taiwan. E-mail: syho@mail.nctu.edu.tw, hsiehch@gmail.com, hayyu.bi93g@nctu.edu.tw.*
- *H.-L. Huang is with the Department of Information Management, Jin Wen Institute of Technology, Hsin-Tien, Taipei, 231 Taiwan. E-mail: hlhuang@jwit.edu.tw.*

Manuscript received 22 Nov. 2005; revised 19 June 2006; accepted 20 Nov. 2006; published online 23 Jan. 2007.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-0135-1105.
Digital Object Identifier no. 10.1109/TCBB.2007.1051.

methods using numerical methods such as steady-state analysis [19] and evolutionary algorithms [10], [11], [12], [13], [14], [16], [17], [18], [21] to solve the optimization problem. A case study considering the estimation of 36 parameters of the nonlinear biochemical dynamic model was investigated by Moles et al. [20] and the result shows that the genetic algorithm (GA) and evolutionary algorithms play an important role in solving the optimization problem of dynamic modeling of genetic networks using the S-system model.

Kikuchi et al. [10] used GA with simplex crossover (SPXGA) to improve the optimization ability for dynamic modeling of genetic networks from $N = 2$ to 5. SPXGA successfully inferred the dynamics of a small genetic network using only the time-series data of gene expression. When dealing with a more complicated structure with a large number of genes (for example, $N = 10$), it is hard to obtain a satisfactory solution in a limited amount of computation time. To infer large-scale genetic network models, Maki et al. [13] proposed an efficient problem decomposition strategy to divide the inference problem into N separated small subproblems. To reduce the search time of the inference problem, Voit and Almeida [18] proposed an approach for transforming the problem to several sets of decoupled algebraic equations, which can be processed efficiently in parallel or sequentially. Kimura et al. [12] used a cooperative coevolutionary algorithm with the problem decomposition strategy on a PC cluster to efficiently infer large-scale S-system models with noisy time-series data. Tsai and Wang [17] used a hybrid differential evolutionary algorithm to obtain the structure of a genetic network and a gradient-based local search to refine the obtained solution. Tominaga and Horton [21] adopted the distributed GA with domain knowledge that the total number of links must equal that expected from a scale-free network.

In this paper, we propose an intelligent two-stage evolutionary algorithm (iTEA) to efficiently infer the S-system models of large-scale genetic networks from small-noise gene expression profiles using a single-processor PC. To cope with the curse of dimensionality, the proposed algorithm consists of two stages, where each uses a divide-and-conquer strategy. The optimization problem is first decomposed into N subproblems having $2(N+1)$ parameters. At the first stage, each subproblem is solved using a novel intelligent genetic algorithm (IGA) that is a specific variant of the intelligent evolutionary algorithm [22]. The intelligent crossover of IGA applies an orthogonal experimental design (OED) [23], [24] to speed up the search by using a systematic reasoning method instead of the conventional generate-and-go method of GA. At the second stage, the obtained N solutions to the N subproblems are combined and refined using an OED-based simulated annealing algorithm (OSA) [25] for handling noisy gene expression profiles. It is shown that OSA performs well in efficiently exploiting the neighborhood of a given initial solution to search for a potentially good approximation of a globally optimal solution [25], [26]. The effectiveness of iTEA is evaluated using simulated expression patterns with and without noise. It will be shown that 1) IGA is efficient enough to solve subproblems, 2) IGA is significantly superior to the existing method SPXGA [10] in solving subproblems, and 3) iTEA performs well in inferring the S-system models of genetic networks from small-noise gene expression profiles.

The remainder of this paper is organized as follows: Section 2 describes the investigated problem and some useful techniques. Section 3 gives the proposed algorithm iTEA. Section 4 gives the performance evaluation of iTEA, including comparisons with the existing methods. Section 5 concludes this paper.

2 The Investigated Problem

2.1 Problem Statement

Generally, the genetic network inference problem using an S-system model is formulated as a parameter optimization problem with $2N(N+1)$ S-system parameters (α_i, β_i, g_{ij}, and h_{ij}) and the following objective function [10], [11], [12]:

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

where $X_{exp,i,t}$ is an experimentally observed expression level of gene i at time t, $X_{cal,i,t}$ is a numerically calculated expression level, N is the number of genes in the network, and T is the number of sampling points of observed data. When all S-system parameters are estimated, $X_{cal,i,t}$ can be derived by using (1) and the given initial level $X_{exp,i,0}$. In the following simulated experiments, the values of $X_{exp,i,t}$ are greater than zero. For real applications, a very small positive value can be assigned to $X_{exp,i,t}$ if some gene expressions are zero at the steady state.

The investigated problem is difficult due to the following characteristics: high degree of freedom, high dimensionality, multimodality, strong interaction among parameters of the S-system model, and measurement noise. Therefore, it is hard to obtain a correct network structure with accurate parameter values. Generally, additional data or biological knowledge is needed to improve solution quality [18].

2.2 Useful Techniques

Two useful techniques in optimizing the objective function (2) are introduced. One is the problem decomposition strategy for large-scale genetic networks [13] and the other is to incorporate a priori knowledge to reduce computation cost [11], [12], [16].

2.2.1 Problem Decomposition

Maki et al. [13] proposed an efficient strategy of dividing the inference problem into N separated small subproblems. Each subproblem corresponds to one gene. The decomposition strategy focuses on the estimation of the $2(N+1)$ S-system parameters $\{\alpha_i, g_{i1}, \dots, g_{iN}, \beta_i, h_{i1}, \dots, h_{iN}\}$ of gene i independently, without involving the estimation of those of other genes at the same time. The objective function of the ith subproblem is given as follows:

$$minimize \quad f_i = \sum_{t=1}^{T}\left(\frac{X_{cal,i,t} - X_{\mathrm{exp},i,t}}{X_{\mathrm{exp},i,t}}\right)^2. \tag{3}$$

To calculate the expression level $X_{cal,i,t}$ of gene i at time t, it additionally needs the expression levels X_j of (1), where $i \neq j$. For noise-free gene expression profiles, the experimentally observed expression levels $X_{exp,j,t}$ of gene j can be directly utilized. To overcome the disadvantage of the problem decomposition when dealing with the given expression levels with large measurement noise (that is, $X_{exp,j,t}$ is not accurate enough) [11], the following modified differential equations are used [11], [12]:

$$\frac{dX_i}{dt} = \alpha_i \prod_{j=1}^{N} Y_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} Y_j^{h_{ij}}, \quad where \begin{cases} Y_j = X_j & if \quad j = i \\ Y_j = \widehat{X}_j & otherwise, \end{cases} \tag{4}$$

where X_j of (1) is replaced with the estimated gene expression level $\widehat{X}_j$. How to effectively obtain accurate $\widehat{X}_j$ is essentially important. Some mathematical methods such as the spline interpolation can be used to estimate the value depending on the size of measurement noise [13]. Kimura et al. [11] used the estimation of the initial gene expression levels in solving individual subproblems for coping with noisy time-series data. Kimura et al. [12] used a cooperative coevolutionary algorithm on a PC cluster to simultaneously solve all subproblems by deriving $\widehat{X}_j$ from estimating the best individuals of the subproblems, each of which is given as a solution of (4). It is shown empirically that the method slightly enhanced the probability of finding the correct interactions of a network [12].

2.2.2 Adding a Penalty Term

In the S-system model, if there is no interaction between two genes i and j, the S-system parameters corresponding to the interaction term, g_{ij} and h_{ij}, are zero. Because the connectivity of the genetic network has been known to be sparse [27], the following fitness function incorporating a penalty term is conveniently added to reduce the search space and improve the accuracy of the inferred genetic network model [11], [12]:

$$minimize \quad f_i = \sum_{t=1}^{T} \left(\frac{X_{cal,i,t} - X_{\exp,i,t}}{X_{\exp,i,t}} \right)^2 + c \sum_{j=1}^{N-I} \left(|G_{ij}| + |H_{ij}| \right), \tag{5}$$

where c is a penalty weight and I is a maximum indegree of the maximal number of genes that directly affect gene i. G_{ij} and H_{ij} are given by rearranging g_{ij} and h_{ij} in ascending order of their absolute values. The penalty term forces most of the kinetic orders (g_{ij}, h_{ij}) down to zero. In the meantime, if the number of genes that directly affect gene i is smaller than I, this term will not be penalized. In such a case, the optimal solutions to the fitness functions (3) and (5) are identical.

To reduce the computation cost, the structure skeletalizing technique [16] was applied. This technique assigns a value of zero to kinetic orders when their absolute values are less than a given threshold δ_s. If a larger value of δ_s is set, the convergence time is rapidly reduced and the false positive rate of the obtained genetic network can be improved, whereas the false negative rate may be increased.

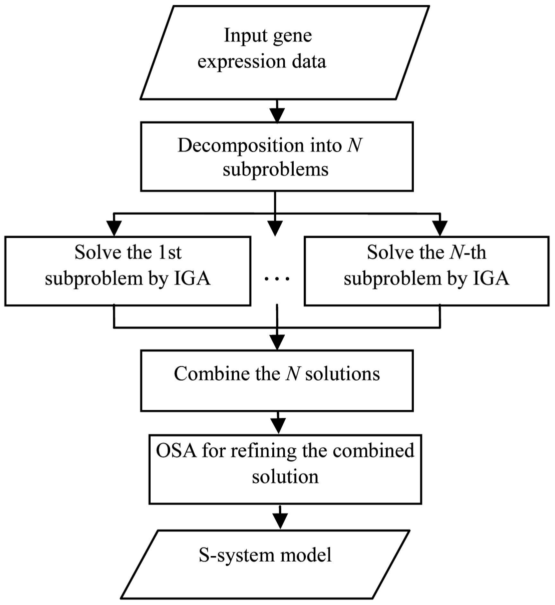
Input gene expression data
Decomposition into N subproblems
Solve the 1st subproblem by IGA
. . .
Solve the N-th subproblem by IGA
Combine the N solutions
OSA for refining the combined solution
S-system model

Fig. 1. Flowchart of the proposed two-stage evolutionary algorithm iTEA.

If a large value of c for the penalty term is used, all values of kinetic orders tend to be small and even zero in the evolutionary process, which results in a relatively large error of expression level fitting. Since the error minimization is the major objective to obtain an accurate structure of genetic network, the value of c cannot be too large. For the simulated small-noise expression profiles in this study, $\delta_s = 3 \times 10^{-2}$ and $c = 1.0$ considering the trade-off. For real-world experiments with measurement noise, it is better to filter small interactions using a larger threshold value, for example, $\delta_s = 0.1$ in [12].

3 Proposed Method

It is well recognized that divide and conquer is an efficient approach in solving large-scale problems. The divide-and-conquer mechanism breaks a large-scale problem into several subproblems that are similar to the original but smaller in size, solves the subproblems concurrently, and then combines these solutions to create a solution to the original problem. Fig. 1 shows a flowchart of the proposed two-stage evolutionary algorithm iTEA. Both IGA and OSA use the divide-and-conquer mechanism based on OED.

IGA is a population-based optimization algorithm that aims at efficiently exploring the whole search space to obtain a potentially good approximation of a global optimum. Therefore, IGA is used to solve each subproblem at the first stage. The N solutions to the N subproblems are combined into an initial solution to the genetic network inference problem. OSA is a point-based optimization algorithm that aims at efficiently exploiting the neighborhood of a given initial solution. In the second stage, OSA used the initial solution obtained by IGA to search for a potentially good approximation of a globally optimal solution to this inference problem.

In the first stage, OSA using a randomly generated initial solution is not efficient enough to obtain satisfactory solutions to N subproblems due to the extremely large and multimodal search space. Therefore, IGA with parallel searching abilities is more efficient than OSA in finding a correct structure from the gene expression data. in the second stage, the number of parameters to be simultaneously optimized is significantly increased from $2(N+1)$ to $2N(N+1)$. By making the best use of the high-quality combined solution, OSA can obtain a satisfactory solution in refining the genetic network using a lesser computation cost than that of IGA. IGA is also available to refine the genetic network if the computation cost is not concerned. Considering noisy gene expression profiles, OSA is more suitable than IGA considering both solution quality and computation cost.

3.1 OED

An efficient way to study the effect of several factors simultaneously is to use OED with both orthogonal array (OA) and factor analysis [23], [24]. The factors are the variables (parameters) that affect the response variables (objective function) and the setting (or the discriminative value) of a factor is regarded as the level of the factor. OED utilizes properties of fractional factorial experiments to efficiently determine the best combination of factor levels to use in design problems. OED specifies the procedure of drawing a representative sample of experiments with the intention of reaching a sound decision [23]. Therefore, OED using OA and factor analysis is regarded as a systematic reasoning method.

OA is a fractional factorial array, which assures a balanced comparison of levels of any factor. In this study, the two-level and three-level OAs are used for IGA and OSA, respectively. The two-level OAs used in IGA are described below. Let there be n factors with two levels each. The total number of level combinations is 2^n for a complete factorial experiment. To use an OA of n factors, we obtain an integer $M = 2^{\lceil \log_2(n+1) \rceil}$, where the bracket represents an upper ceiling operation, build an OA $\mathrm{L}_M(2^{M-1})$ with M rows and $M-1$ columns, use the first n columns, and ignore the other $M-n-1$ columns. OA can reduce the number of level combinations for factor analysis. The number of OA combinations required to analyze all individual factors is only $M = O(n)$, where $n+1 \le M \le 2n$.

OSA uses three-level OAs, where each factor has three levels. The total number of level combinations is 3^n for a complete factorial experiment. To use a three-level OA of n factors, we obtain an integer $M = 3^{\lceil \log_3(2n+1) \rceil}$, build an OA $\mathrm{L}_M(3^{(M-1)/2})$ with M rows and $(M-1)/2$ columns, use the first n columns, and ignore the other $(M-1)/2-n$ columns. The number of OA combinations required to analyze all individual factors is only $M = O(n)$, where $2n+1 \le M \le 6n-3$. The algorithm for constructing the two and three-level OAs can be found in [25]. After proper tabulation of experimental results, the summarized data are analyzed using factor analysis to determine the relative level effects of factors.

Factor analysis using the OA's tabulation of experimental results can evaluate the effects of individual factors on the evaluation function, rank the most effective factors, and determine the best level for each factor such that the evaluation function is optimized. Consider the OA $\mathrm{L}_M(2^{M-1})$ or $\mathrm{L}_M(3^{(M-1)/2})$ used. Let y_t denote a function value of the combination t, where $t = 1, \dots, M$. Define the main effect of factor d with level k as S_{dk}, where $d = 1, \dots, n$:

$$S_{dk} = \sum_{t=1}^{M} y_t \cdot W_t, \tag{6}$$

where $W_t = 1$ if the level of factor d of combination t is k; otherwise, $W_t = 0$. Consider that the objective function is to be minimized. For the two-level OA, level 1 of factor d makes a better contribution to the objective function than level 2 of factor d does when $S_{d1} < S_{d2}$. If $S_{d1} > S_{d2}$, level 2 is better. If $S_{d1} = S_{d2}$, levels 1 and 2 have the same contribution. The main effect reveals the individual effect of a factor. The most effective factor d has the largest main effect difference, $MED_d = |S_{d1} - S_{d2}|$.

For the three-level OA, the level k of factor d makes a better contribution to the objective function than the other two levels of factor d do when $S_{dk} = \min\{S_{d1}, S_{d2}, S_{d3}\}$. On the contrary, if the objective function is to be maximized, the level k is the best when $S_{dk} = \max\{S_{d1}, S_{d2}, S_{d3}\}$. The most effective factor has the largest main effect difference, $MED_d = \max\{S_{d1}, S_{d2}, S_{d3}\} - \min\{S_{d1}, S_{d2}, S_{d3}\}$. After the better (best) of the two (three) levels of each factor is determined, a reasoned combination consisting of n factors with the better (best) levels can be easily derived.

3.2 IGA for Solving Subproblems

3.2.1 Intelligent Crossover

The intelligent crossover plays an important role in IGA. IGA solves an individual subproblem with N genes having $2(N+1)$ parameters to be optimized. The intelligent crossover uses a divide-and-conquer approach which consists of adaptively dividing two parents into n pairs of parameter groups, economically identifying the potentially better of the two groups of each pair, and systematically obtaining a potentially good approximation of the best among all 2^n combinations using at most $2n$ fitness evaluations. Like traditional GAs, two parents, P_1 and P_2, produce two children, C_1 and C_2, using one crossover operation. The intelligent crossover determines the recombination of P_1 and P_2 for efficiently generating good children. Let the set of parameters in the ith subproblem be $\{\alpha_i, g_{i1}, \dots, g_{iN}, \beta_i, h_{i1}, \dots, h_{iN}\}$. We divided the two sets, $\mathrm{INC} = \{\alpha_i, g_{i1}, \dots, g_{iN}\}$ and $\mathrm{DEC} = \{\beta_i, h_{i1}, \dots, h_{iN}\}$, that control the gene expression level, increasing or decreasing into $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$ groups, respectively. To make sufficient use of all columns in OAs, n is usually set to $2^\omega - 1$, where ω is an integer. In this study, we used $n = 7$ for problems with $N \le 30$. The value of n would properly increase when N increases. The discussion between n and the number of parameters to be optimized can be found in [22].

Because the parameters belonging to the same one or two sets INC and DEC have strong interactions, we do not use the conventional encoding scheme of GA that all parameters are encoded into a chromosome in a fixed order. Instead, all parameters are represented using real values with no order. For each time, using an intelligent crossover operation, INC and DEC are randomly divided into $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$ groups with a variable size for each group. The parameters of two parents are grouped using the same division operation. Each group is treated as a factor. The

TABLE 1
An Illustrative Example of an Intelligent Crossover Using OA $L_8(2^7)$

Combination t	Factor d							y_t	Rank
	1	2	3	4	5	6	7		
1	1	1	1	1	1	1	1	40.12	81/128
2	1	1	1	2	2	2	2	39.70	72/128
3	1	2	2	1	1	2	2	36.34	23/128
4	1	2	2	2	2	1	1	41.35	97/128
5	2	1	2	1	2	1	2	41.40	102/128
6	2	1	2	2	1	2	1	42.60	125/128
7	2	2	1	1	2	2	1	34.32	13/128
8	2	2	1	2	1	1	2	35.65	18/128
S_{d1}	157.50	163.81	149.78	152.17	154.70	158.51	158.39		
S_{d2}	153.97	147.65	161.68	159.30	156.76	152.95	153.08		
MED_t	3.54	16.16	11.90	7.13	2.06	5.56	5.31		
Child 1 (C_1)	2	2	1	1	1	2	2	30.22	2/128
Child 2 (C_2)	2	2	1	1	2	2	2	30.80	4/128

TABLE 2
The Contents of Parents and Children

	α_1	g_{11}	g_{12}	g_{13}	g_{14}	g_{15}	β_1	h_{11}	h_{12}	h_{13}	h_{14}	h_{15}	y
P_1	2.98	2.08	0.98	-1.17	2.13	0.00	2.71	2.04	0.10	1.89	2.33	-0.06	40.12
P_2	12.68	0.58	-0.24	0.79	0.33	0.82	10.69	0.74	2.31	3.00	1.33	-1.33	32.68
C_1	12.68	0.58	-0.24	0.79	0.33	0.82	2.71	2.04	0.10	1.89	2.33	-0.06	30.22
C_2	12.68	0.58	-0.24	0.79	0.33	0.82	2.71	0.74	2.31	1.89	2.33	-0.06	30.80

n factors are randomly numbered using OED. The numbering order does not affect the effectiveness of the intelligent crossover because of the property of OA. Note that there is no fixed genotype of the S-system parameters used. The following steps describe how to use OED with n factors to achieve the intelligent crossover of IGA for a fitness function y:

Step 1. The two sets $\{\alpha_i, g_{i1}, \ldots, g_{iN}\}$ and $\{\beta_i, h_{i1}, \ldots, h_{iN}\}$ of S-system parameters are randomly divided into $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$ groups (factors), respectively.

Step 2. Use a two-level OA $L_{n+1}(2^n)$ with $n+1$ rows and n columns.

Step 3. Let levels 1 and 2 of factor d represent the dth groups coming from parents P_1 and P_2, respectively.

Step 4. Evaluate the fitness values y_t for experiment t, where $t = 2, \ldots, n+1$. The value y_1 is the fitness value of P_1.

Step 5. Compute the main effect S_{dk}, where $d = 1, \ldots, n$ and $k = 1, 2$.

Step 6. Determine the better of the two levels of each factor according to the main effect.

Step 7. Form the chromosome of C_1 using the combination of the better groups from the derived corresponding parents.

Step 8. Form the chromosome of C_2 similarly to C_1, but, for the factor with the smallest main effect difference, adopt the other level.

Step 9. Use the best two individuals among P_1, P_2, C_1, C_2, and n combinations of OA as the final children C_1 and C_2 for elitist strategy.

One intelligent crossover operation takes $n+2$ fitness evaluations to explore the search space of 2^n combinations. Generally, C_1 is a potentially good approximation of the best among all 2^n combinations.

3.2.2 Illustrative Example of Intelligent Crossover

Tables 1 and 2 show an illustrative example of using intelligent crossover with OED in solving the first subproblem of inferring an S-system model with $N = 5$. The details of the test problem are given in Section 4.1. We used an OA $L_8(2^7)$ for $n = 7$. The two sets of S-system parameters, $\{\alpha_1, g_{11}, \ldots, g_{15}\}$ and $\{\beta_1, h_{11}, \ldots, h_{15}\}$, are randomly divided and assigned to four and three groups (factors), respectively, as follows: $V_1 = \{h_{13}, h_{15}\}$, $V_2 = \{g_{14}\}$, $V_3 = \{g_{12}, g_{13}\}$, $V_4 = \{\alpha_1, g_{15}\}$, $V_5 = \{h_{11}, h_{12}\}$, $V_6 = \{\beta_1, h_{14}\}$, and $V_7 = \{g_{11}\}$. The parameter values of the parents are given in Table 2. Table 1 shows all of the results of the intelligent crossover using OED. First, we evaluate the response variable y_t of the combination t, where $t = 1, 2, \ldots, 8$. Second, we compute the main effect S_{dk}, where $d = 1, 2, \ldots, 7$ and $k = 1, 2$. For example, $S_{22} = y_3 + y_4 + y_7 + y_8 = 147.65$. Third, the better level of each factor based on the main effect is determined. For example, the better level of factor 1 is level 2 since $S_{12}(153.97) < S_{11}(157.50)$. Finally, the better levels of factors (V_1, V_2, V_3, V_4, V_5, V_6, V_7) are (2, 2, 1, 1, 1, 2, 2) and, then, $y = 30.22$ can be obtained from the reasoned combination. This reasoned combination is used to form the child C_1 of the crossover operation. The least effective factor is $d = 5$ with $MED_5 = 2.06$, which is the smallest, so the second child C_2 is formed similarly to C_1 except that V_5 adopts level 2. Note that the ranks of C_1 and C_2 are 2 and 4, respectively, among the 128 combinations of a complete factorial experiment. It reveals that the reasoning operation of an intelligent crossover for generating children is efficient.

3.2.3 The Used IGA

IGA is used to solve the ith individual subproblems with the fitness function (5). The gene expression level of $\widehat{X}_j$ in (4) is obtained from $X_{exp,j,t}$ without using additional estimation methods based on the following reasons:

1. According to the simulation using IGA, the method of directly using $X_{exp,j,t}$ is simple and fast and its solution is accurate enough in terms of the fitness value from noise-free gene expression profiles.
2. We would further refine the combined solutions of the N subproblems from the aspect of global optimization using OSA for handling small-noise gene expression profiles.
3. The estimation method for $\widehat{X}_j$ using a cooperative coevolutionary algorithm on a PC cluster [12] is not suitable for the IGA-based method because IGA solves each subproblem independently on a single-processor PC. Furthermore, the estimation method only slightly enhanced the probability of finding the correct interactions of a network for large-noise gene expression profiles [12].

The main differences between the used IGA and the conventional GAs are chromosome encoding and the crossover operation mentioned above. Besides, the used mutation is also different from the conventional one, described as follows: Assume a real-value parameter x is to be mutated. A perturbation $\overline{x}$ is generated by the Cauchy-Lorentz probability distribution [28]. The mutated value of x is $x' = x + \overline{x}$ or $x - \overline{x}$, determined randomly. If x' is out of the domain range of x, a random value is assigned to x'. The used simple IGA is described below:

Step 1 (Initiation). Randomly generate an initial population with N_{pop} feasible individuals of $2(N+1)$ real-value parameters.

Step 2 (Evaluation). Evaluate the fitness values of all individuals.

Step 3 (Selection). Use the simple truncation selection that replaces the worst $P_{\text{s}} \times N_{\text{pop}}$ individuals with the best $P_{\text{s}} \times N_{\text{pop}}$ individuals to form a new population, where P_{s} is a selection probability. Let I_{best} be the best individual in the population.

Step 4 (Crossover). Randomly select $P_{\text{c}} \times N_{\text{pop}}$ individuals, including I_{best}, where P_{c} is a crossover probability. Perform intelligent crossover operations for all selected pairs of parents.

Step 5 (Mutation). Apply the abovementioned mutation operator to the population using a mutation probability P_{m}. To prevent the best fitness value from deteriorating, mutation is not applied to the best individual.

Step 6 (Termination test). If a prespecified number N_{eval} of fitness evaluations is achieved or some stopping condition is met, then stop the algorithm. Otherwise, go to Step 2.

3.3 OSA for Refining the Combined Solution

To compensate for the disregard of estimating accurate gene expression levels of other genes from the noisy data of gene expression, all of the solutions to the N subproblems are combined and then refined using OSA from the aspect of global optimization. The main difference between OSA and the conventional simulated annealing is the move generation mechanism. OSA uses an intelligent generation mechanism (IGM) based on OED to systematically reason out a good candidate solution as the next move. The high performance of OSA arises from IGM, which is the main phase of OSA. IGM is similar to the intelligent crossover of IGA using the divide-and-conquer mechanism for large-scale optimization problems, which is also efficient in determining a good approximation of the best solution in the neighborhood of the current solution. OSA uses the following objective function for global optimization:

$$minimize$$
$$F = \sum_{i=1}^{N}\sum_{t=1}^{T}\left(\frac{X_{cal,i,t} - X_{exp,i,t}}{X_{exp,i,t}}\right)^2 + c\sum_{i=1}^{N}\sum_{j=1}^{N-I}(|G_{ij}| + |H_{ij}|). \tag{7}$$

3.3.1 IGM

Let all of the N solutions be combined into an initial solution S of OSA to be refined. Let $S = (s_1, \dots, s_p)$, where s_i is one of the S-system parameters and $p = 2N(N+1)$. IGM generates two temporary solutions, $S_{\text{A}} = (a_1, \dots, a_p)$ and $S_{\text{B}} = (b_1, \dots, b_p)$, by perturbing S, where a_i and b_i are defined as follows:

$$a_i = s_i + s'_i; b_i = s_i - s'_i, i = 1, \dots, p. \tag{8}$$

The values of s'_i are generated by the Cauchy-Lorentz probability distribution. IGM aims at efficiently combining the good values of parameters from solutions S, S_{A}, and S_{B} to generate a good candidate solution Q for the next move of S.

Divide all of the p parameters into m nonoverlapping groups with variable sizes, using the same division operation for S, S_{A}, and S_{B}. In this study, the used OA is $L_{2m+1}(3^m)$ and $m = 13$ for $N \leq 30$. How to decide the proper value of m and OA can be found in [25]. Due to the same reason as that of intelligent crossover, the two sets $\{\alpha_i, g_{ij}\}$ and $\{\beta_i, h_{ij}\}$ are randomly divided into seven and six parameter groups, respectively. How to perform an IGM operation with m groups using a current solution S and the objective function F in (7) is described as follows:

Step 1. Generate two temporary solutions S_{A} and S_{B} using S.

Step 2. Using the same division operation, randomly divide the two sets $\{\alpha_i, g_{i1}, \dots, g_{iN}\}$ and $\{\beta_i, h_{i1}, \dots, h_{iN}\}$ in S, S_{A}, and S_{B} into $(m+1)/2$ and $(m-1)/2$ groups (factors), respectively.

Step 3. Let levels 1, 2, and 3 of a factor represent the groups coming from S, S_{A}, and S_{B}, respectively.

Step 4. Compute y_t of the generated combination, where $t = 2, \dots, 2m+1$. Note that $y_1 = F(S)$.

Step 5. Compute the main effect S_{dk}, where $d = 1, \dots, m$ and $k = 1, 2, 3$.

Step 6. Determine the best of the three levels of each factor according to the main effect.

Step 7. Form the candidate solution Q using the combination of the best groups.

Step 8. Verify that Q is superior to the $2m$ sampling solutions derived from the OA combinations and $Q \neq S$. If it is not true, select the best from the $2m$ sampling solutions as Q.

TABLE 3
The S-System Parameters of a Small-Scale Target Network with $N = 5$ from [10]

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	1	0	-1	10	2	0	0	0	0
2	10	2	0	0	0	0	10	0	2	0	0	0
3	10	0	-1	0	0	0	10	0	-1	2	0	0
4	8	0	0	2	0	-1	10	0	0	0	2	0
5	10	0	0	0	2	0	10	0	0	0.	0	2

The number of objective function evaluations is $2m + 1$ per IGM operation, which includes $2m$ evaluations in Step 4 and one in Step 8.

3.3.2 Global Optimization Using OSA

The main power of OSA mainly arises from using IGM to efficiently search for a good candidate solution. OSA uses a simple geometric cooling rule by updating the temperature at the $(i+1)\text{th}$ temperature step using the formula: $Temp_{i+1} = CR \cdot Temp_i, i = 0, 1, \ldots,$ where CR is the cooling rate, which is a constant smaller than 1 but close to 1 (for example, $CR = 0.99$). The higher the temperature, the larger the possibility of accepting a candidate solution worse than the current solution is. In this study, a simple version of OSA proposed in [25] is used. The OSA used is described below:

Step 1 (Initialization). Initialize $Temp = Temp_0$ and CR. Let the combined solution S be an initial solution and compute $F(S)$.

Step 2 (Perturbation). Perform an IGM operation using S to generate a candidate solution Q.

Step 3 (Acceptance criterion). Accept Q to be the new S with probability $P(Q)$:

$$P(Q) = \begin{cases} 1, & if \quad F(Q) \le F(S) \\ \exp(\frac{F(S)-F(Q)}{Temp}), & if \quad F(Q) > F(S). \end{cases} \tag{9}$$

Step 4 (Decreasing temperature). Let the new value of $Temp$ be $CR \cdot Temp$.

Step 5 (Termination test). If a prespecified stopping criterion is met, stop the algorithm. Otherwise, go to Step 2.

In this study, if the solutions to the subproblems are accurate enough and whose fitness values are sufficiently small, OSA plays a role in finely tuning the values of parameters but not the structure from the aspect of global optimization. In such a case, $Temp_0$ can be set to a very small value. If the fitness values are not satisfactory, $Temp_0$ can be enlarged to search for a better solution in which the structure of the S-system model may be modified.

3.4 iTEA Using IGA and OSA

The proposed algorithm, iTEA, uses both IGA and OSA in Stages 1 and 2, respectively. IGA aims to obtain solutions to subproblems with significant accuracy in terms of the objective function value that can best fit the given gene expression profiles. If the noise is very small, IGA is effective enough and the improvement of OSA in Stage 2 is not significant. When the noise becomes larger, the best fit of the observed gene expression profiles is left to OSA from the aspect of global optimization. The proposed iTEA is given as follows:

Stage 1. Apply IGA to solve N individual subproblems independently using the fitness function (5). $R > 1$ independent runs are conducted for each subproblem and the best solution to each subproblem is selected. In this study, $R = 10$ is suggested.

Stage 2. Combine these N best solutions $(\alpha_i, g_{i1}, \ldots, g_{iN}, \beta_i, h_{i1}, \ldots, h_{iN})$, $i = 1, \ldots, N$, into an initial solution S. Apply OSA to refine S using the objective function F in (7).

4 Experimental Results

We conducted some evaluations for the proposed algorithm iTEA. The parameters used in iTEA are described below. For IGA, $N_{\text{pop}} = 20$, $P_{\text{s}} = 0.2$, $P_{\text{c}} = 0.8$, and $P_{\text{m}} = 0.2$. The penalty coefficient $c = 1.0$ is in (5) and (7). From our computer simulation results, OSA generally performs well by giving a very small value to $Temp_0$, for example, 0.001.

4.1 Experiment 1—Performance of IGA

In this experiment, two target genetic networks with $N = 5$ and 10 are used to evaluate the performance of IGA. For a small-scale target network, we used the same S-system model of a genetic network consisting of $N = 5$ genes as that in [10]. This model has been developed to analyze the interaction of regulator and effector genes, which has feedback loops. The S-system parameters of the target network are given in Table 3. The search regions of the S-system parameters are $[0, 15.0]$ for α_i and β_i and $[-3.0, 3.0]$ for g_{ij} and h_{ij}. To enhance the probability of finding the correct solution, 15 sets of noise-free time-series data were used, each covering all five genes, as a sufficient amount of observed gene expression profiles. The sets of time-series data were obtained by using (1). The initial values of these sets are listed in Table 4. $T = 11$ sampling points for the time-series data were assigned on each gene in each set and, hence, the observed time-series data on each gene consist of $15 \times 11 = 165$ sampling points.

For this network model, we have to estimate 60 parameters of the S-system. Let $I = 2$ and $N_{\text{eval}} = 2 \times 10^5$. For each run, the total fitness value of $N = 5$ subproblems using the F function of (7) is recorded. For evaluating the robustness of IGA, 30 independent runs of IGA were conducted. The best and mean values of F from 30 runs are 0.00171 and 0.29267, respectively. Table 5 shows the S-system parameters of the best solutions to the five subproblems in terms of fitness value using (5), where the

TABLE 4
Fifteen Sets of Initial Gene Expression Levels of the Target Network with $N = 5$ from [10]

Set	X_1	X_2	X_3	X_4	X_5
1	0.70	0.12	0.14	0.16	0.18
2	0.10	0.70	0.14	0.16	0.18
3	0.10	0.12	0.70	0.16	0.18
4	0.10	0.12	0.14	0.70	0.18
5	0.10	0.12	0.14	0.16	0.70
6	0.70	0.70	0.14	0.16	0.70
7	0.10	0.70	0.70	0.16	0.18
8	0.10	0.12	0.70	0.70	0.18
9	0.10	0.12	0.14	0.70	0.70
10	0.70	0.12	0.14	0.16	0.70
11	0.20	0.70	0.14	0.10	0.40
12	0.10	0.15	0.10	0.16	0.18
13	0.30	0.12	0.70	0.10	0.10
14	0.10	0.25	0.70	0.10	0.30
15	0.20	0.12	0.70	0.16	0.20

TABLE 5
The Estimated S-System Parameter Sets with $N = 5$

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}
1	5.002163	0	0	0.999995	0	-0.999868
2	10.000714	1.999865	0	0	0	0
3	9.997750	0	-1.000208	0	0	0
4	7.991448	0	0	2.000909	0	-1.000599
5	10.004309	0	0	0	1.998261	0

i	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	10.001932	1.996719	0	0	0	0
2	9.996852	0	1.999167	0	0	0
3	10.000212	0	-1.000058	1.999683	0	0
4	9.992125	0	0	0	2.006558	0
5	9.976999	0	0	0	0	1.994196

The fitness value of each subproblem is smaller than 10^{-6}.

TABLE 6
S-System Parameters of the Target Network with $N = 10$

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	g_{i6}	g_{i7}	g_{i8}	g_{i9}	g_{i10}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}	h_{i6}	h_{i7}	h_{i8}	h_{i9}	h_{i10}
1	5	0	0	0	1	0	-2	0	0	0	0	10	2	0	0	0	0	0	0	0	0	0
2	10	0	0	1	0	0	0	0	-1	0	0	10	0	2	0	0	0	0	0	0	0	0
3	8	-1	0	0	-1	0	0	0	0	0	0	10	0	0	2	0	0	0	0	0	0	0
4	10	0	0	0	0	2	0	0	0	1	0	10	0	0	0	2	0	0	0	0	0	0
5	10	0	2	0	0	0	-1	0	0	0	0	10	0	0	0	0	2	0	0	0	0	0
6	5	0	0	0	0	0	0	0	0	2	-2	10	0	0	0	0	0	2	0	0	0	0
7	10	0	0	0	0	0	1	0	0	0	-1	10	0	0	0	0	0	0	2	0	0	0
8	5	1	-2	0	0	0	0	1	0	0	0	10	0	0	0	0	0	0	0	2	0	0
9	10	0	0	1	0	0	0	0	-2	0	0	10	0	0	0	0	0	0	0	0	2	0
10	8	2	0	0	0	0	0	-1	0	0	0	10	0	0	0	0	0	0	0	0	0	2

fitness value of each subproblem is smaller than 10^{-6}. The genetic structure is correct and the parameter values are precise enough to biologically interpret the network. IGA running on a single-processor PC (Pentium III 933 MHz) takes 5.8 minutes on average to solve five subproblems in an independent run. Because the fitness values of the solutions obtained from Stage 1 are small enough, no refinement using OSA is required. Therefore, iTEA with $R = 10$ takes 58 $(= 5.8 \times 10)$ minutes.

Another genetic network of $N = 10$ genes is given in Tables 6 and 7. There are 169 and 51 zero and nonzero values of S-system parameters, respectively. We used 15 sets of gene expression profiles with $I = 3$ and $T = 11$. The stopping condition is when the fitness value is still not effectively improved after a constant number $N_{\text{gen}} = 100$ of generations. The best solutions in terms of the fitness value from 30 independent runs are listed in Table 8. The corresponding fitness values using (5) for the 10 subproblems are given in Table 9. There are no false-negative interaction and only nine false-positive interactions whose magnitudes are relatively small. This experiment running on a single-processor PC (Pentium 4 2.8 GHz) takes 45 minutes on the average to solve 10 subproblems.

4.2 Experiment 2—Comparison between SPXGA and IGA

In this experiment, we conducted some experiments using S-system models containing $N = 5, 10, \ldots, 30$ genes as target networks to show that IGA is significantly better

TABLE 7
Fifteen Sets of Initial Gene Expression Levels of the Target Network with $N = 10$

Set	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
1	0.10	0.12	0.14	0.16	0.18	0.20	0.22	0.24	0.26	0.28
2	0.70	0.12	0.14	0.16	0.18	0.20	0.22	0.24	0.26	0.70
3	0.10	0.70	0.14	0.16	0.18	0.20	0.22	0.24	0.70	0.28
4	0.10	0.12	0.70	0.16	0.18	0.20	0.22	0.70	0.26	0.28
5	0.10	0.12	0.14	0.70	0.18	0.20	0.70	0.24	0.26	0.28
6	0.10	0.12	0.14	0.16	0.70	0.20	0.22	0.24	0.26	0.28
7	0.10	0.12	0.14	0.16	0.18	0.70	0.22	0.24	0.26	0.28
8	0.10	0.12	0.14	0.16	0.18	0.20	0.70	0.24	0.26	0.28
9	0.10	0.12	0.14	0.16	0.18	0.20	0.22	0.70	0.26	0.28
10	0.10	0.12	0.14	0.16	0.18	0.20	0.22	0.24	0.70	0.28
11	0.14	0.12	0.14	0.16	0.70	0.20	0.22	0.24	0.26	0.10
12	0.45	0.10	0.14	0.16	0.10	0.10	0.22	0.70	0.26	0.10
13	0.20	0.12	0.14	0.10	0.18	0.10	0.70	0.24	0.70	0.10
14	0.50	0.10	0.14	0.16	0.18	0.20	0.70	0.70	0.26	0.10
15	0.30	0.12	0.10	0.16	0.18	0.20	0.22	0.24	0.70	0.10

TABLE 8
The Obtained S-System Parameters with $N = 10$

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	g_{i6}	g_{i7}	g_{i8}	g_{i9}	g_{i10}
1	5.00	0	0	0	1.00	0	-2.00	0	0	0	0
2	9.94	0	0	1.00	0	0	0	0	-1.00	0	0
3	8.02	-1.00	0	0	-1.00	0	0	0	0	0	0
4	9.99	0	0	0	0	2.00	0	0	0	1.00	0
5	10.00	0	2.00	0	0	0	-1.00	0	0	0	0
6	5.01	0	0	0	0	0	0	0	0	2.00	-1.99
7	10.06	0	0	0	0	0	1.00	0	0	0	-1.00
8	4.98	1.00	-1.99	0	0	0	0	1.00	0	0	0
9	9.41	0	0	1.02	0	0	0	0	-2.02	$\boxed{-0.03}$	0
10	8.00	2.00	0	0	0	0	0	-1.00	0	0	0

i	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}	h_{i6}	h_{i7}	h_{i8}	h_{i9}	h_{i10}
1	10.00	2.00	0	0	0	0	0	0	0	0	0
2	9.22	0	2.05	0	0	0	0	$\boxed{-0.03}$	0	0	$\boxed{-0.06}$
3	10.07	0	0	2.00	0	0	0	0	0	0	0
4	10.00	0	0	0	2.00	0	0	0	0	0	0
5	9.98	0	0	0	0	2.00	0	0	0	0	0
6	9.39	0	0	0	0	0	2.02	0	0	0	0
7	10.09	0	0	0	0	0	$\boxed{-0.05}$	2.04	0	$\boxed{0.03}$	0
8	11.59	0	$\boxed{0.30}$	0	0	0	0	0	1.96	$\boxed{-0.28}$	0
9	9.09	$\boxed{0.04}$	0	0	0	0	0	0	$\boxed{0.13}$	2.13	0
10	10.01	0	0	0	0	0	0	0	0	0	2.00

The nine highlighted numbers indicated false-positive interactions. No false-negative interaction is found.

than SPXGA [10] for solving subproblems in terms of fitness value using the same number N_{eval} of fitness evaluations. We generated feasible expression patterns for comparing the optimization abilities of IGA and SPXGA. Six sets of time-series data of gene expression are generated, each covering all N genes. The values of gene expression levels are generated in the range $[0, 1.0]$. The search regions were $[0, 15.0]$ for α_i and β_i and $[-3.0, 3.0]$ for g_{ij} and h_{ij}. Let $I = 3$ and $T = 11$. We conducted 30 independent runs for the first subproblem of each experiment using IGA and SPXGA to compare the effectiveness by a statistical t-test method. For each experiment, the stopping condition is $N_{\text{eval}} = 5,000 \times N$. The parameters used in SPXGA are identical to those in [10].

Fig. 2 gives the convergences of the 30 runs for comparisons between IGA and SPXGA with various values of N. It can be seen that IGA is obviously superior to SPXGA, especially when N is large. Table 10 shows the t-test results of the six target networks. It can be found that the mean fitness values and variances of IGA are much smaller than those of SPXGA, where their P value is near to zero. These results reveal that IGA is significantly better than SPXGA in solving the individual subproblems in a limited amount of computation time.

TABLE 9
The Fitness Values Using (5) for All Subproblems with $N = 10$

i	1	2	3	4	5
f_i	0.000001	0.000416	0.000019	0.000001	0.000004
i	6	7	8	9	10
f_i	0.003180	0.000547	0.000020	0.001786	0.000000

4.3 Experiment 3—iTEA for Noisy Gene Expression Profiles

In this experiment, we will evaluate the proposed iTEA using noisy gene expression profiles. We adopted the Gaussian noise, which is commonly used for the simulated experiment [29]. Four target genetic networks with $N = 5$, 10, 15, and 30 are used, where the networks with $N = 5$ and 10 are the same as those in Experiment 1. The target

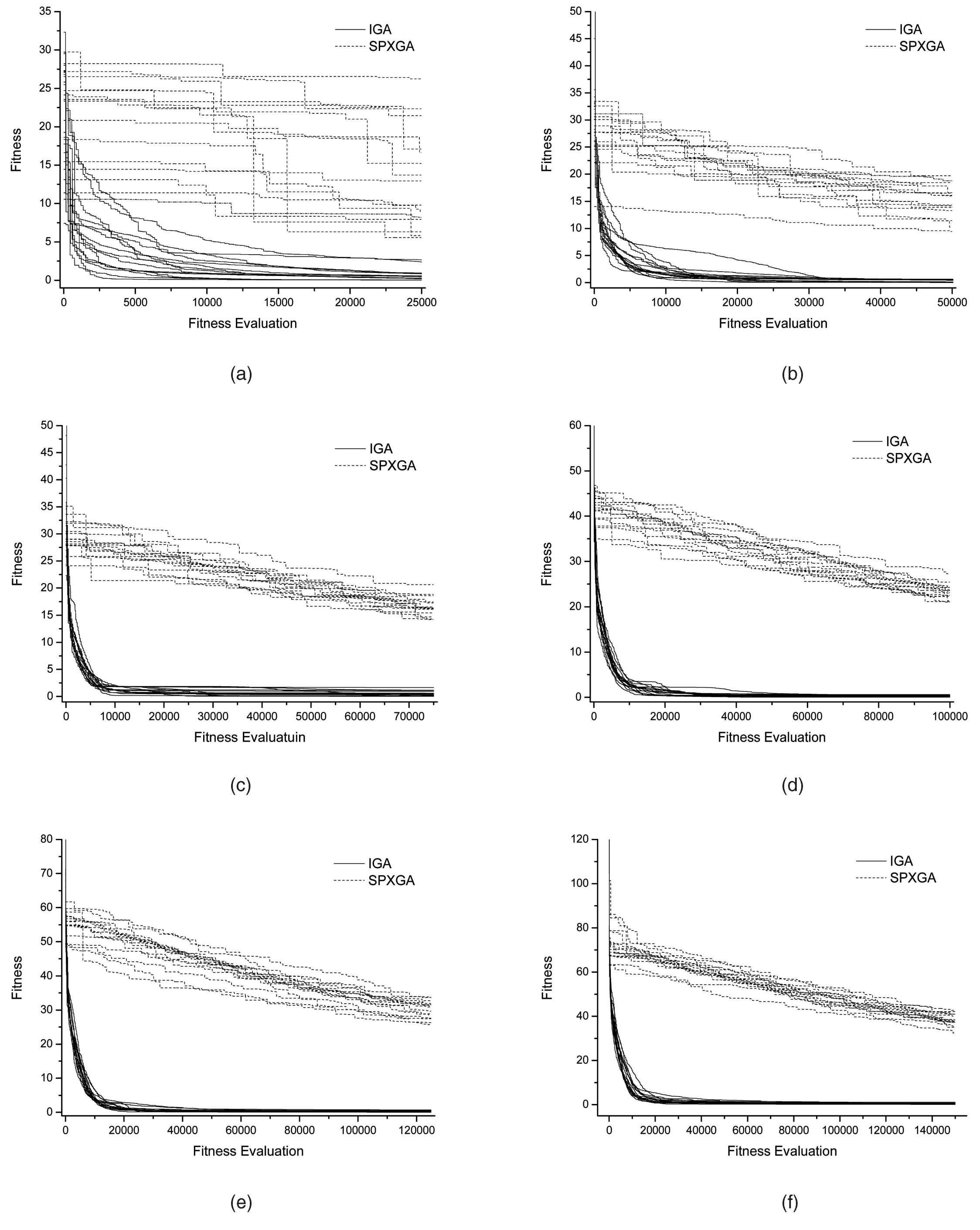

Fig. 2. The convergence comparison between IGA and SPXGA using 30 independent runs. (a) $N = 5$. (b) $N = 10$. (c) $N = 15$. (d) $N = 20$. (e) $N = 25$. (f) $N = 30$.

network of $N = 15$ has also 15 sets of gene expression profiles. These three networks with $N = 5$, 10, and 15 adopted $I = 3$ and $T = 11$. The S-system parameters of the network with $N = 30$ are obtained from [12]. The network with $N = 30$ has 20 sets of gene expression profiles, $I = 5$, and $T = 11$. The values of the gene expression levels were generated in the range $[0, 2.0]$. The search regions were $[0, 3.0]$ for α_i and β_i and $[-3.0, 3.0]$ for g_{ij} and h_{ij}. We added k percent Gaussian noise to all gene expression level points of N genes. The mean of the Gaussian noise is zero and the standard deviation equals to $X_{exp,i,t} \times k$ percent.

We applied iTEA to estimate the parameters of the S-system model. The function value of (7) reflecting the fitting quality of the time-series data is used to evaluate the ability of the used optimization algorithm. However, the major concern is to obtain a correct network structure with accurate parameter values. We define the true-positive rate as sensitivity $SN = TP/(TP + FN)$, where TP is true positive and FN is false negative, and the true negative rate as specificity $SP = TN/(TN + FP)$, where TN is true negative and FP is false positive.

TABLE 10
T-Test Results for Comparisons between IGA and SPXGA with Various Values of N

N	5		10		15	
N_{eval}	25000		50000		75000	
Algorithm	IGA	SPXGA	IGA	SPXGA	IGA	SPXGA
Best	0.01578	4.65101	2.84×10-4	9.42883	0.02343	13.30884
Worst	2.65831	26.24824	0.72764	21.09678	1.64378	20.64385
Mean	0.53829	11.41142	0.24137	15.24435	0.49844	16.3107
Variance	0.4979	29.68118	0.06484	10.13658	0.18244	2.97019
t value	-11.25817		-25.56804		-47.24382	
p value	4.20046×10^{-12}		1.92315×10^{-21}		5.45875×10^{-29}	

N	20		25		30	
N_{eval}	100000		125000		150000	
Algorithm	IGA	SPXGA	IGA	SPXGA	IGA	SPXGA
Best	0.01155	18.86678	0.01924	25.71603	0.06302	32.38494
Worst	1.00323	27.30871	0.76638	34.00673	1.1908	44.98229
Mean	0.30912	22.65516	0.32504	30.39126	0.58418	38.86353
Variance	0.05177	3.27567	0.05363	6.44214	0.11128	7.25008
t value	-64.91653		-65.25943		-76.31632	
p value	5.91115×10^{-33}		5.07875×10^{-33}		5.56666×10^{-35}	

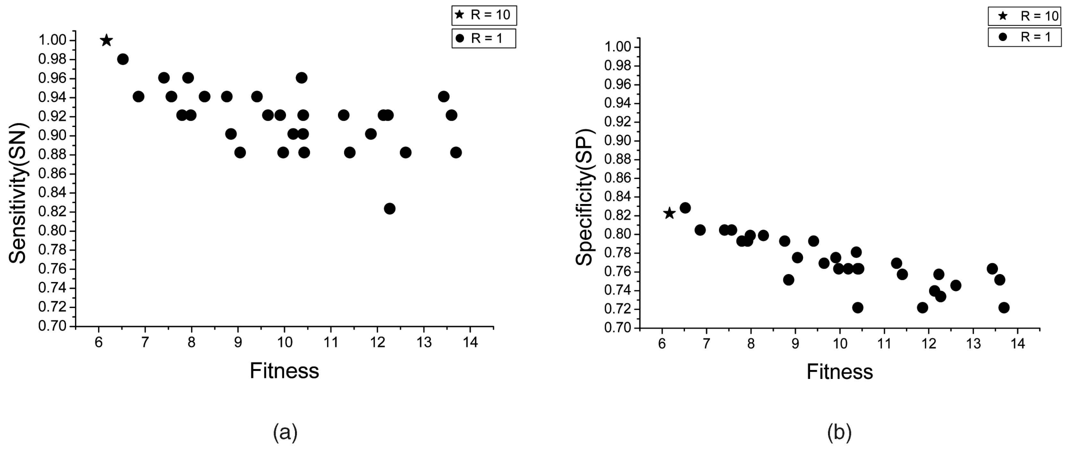

Fig. 3. The distribution of the 30 solutions with $R = 1$ and one solution with $R = 10$ using IGA only, without refinement of OSA, from gene expression profiles of $N = 10$ and 5 percent Gaussian noise. (a) Fitness value versus sensitivity. (b) Fitness value versus specificity.

To illustrate the effectiveness of $R = 10$ runs in Stage 1 and refinement using OSA in Stage 2, we take the target model of $N = 10$ with 5 percent Gaussian noise as an example. At first, IGA is performed in one run ($R = 1$) to solve each individual subproblem and, then, all of the $N = 10$ solutions to the 10 subproblems are combined as a final solution to the inference problem. On the other hand, a combined solution S is also obtained from Stage 1 using IGA with $R = 10$. Fig. 3 shows the distribution of S and the 30 final solutions. In Fig. 3, it can be found that the best solution S_{best} of $R = 1$ has values $F = 6.42$, $SN = 98\%$, and $SP = 82.7\%$. The solution S has $F = 6.17$, $SN = 100\%$, and $SP = 82.25\%$, which is slightly better than S_{best}. Therefore, it is effective to combine the best of the $R = 10$ solutions for all individual subproblems. After performing OSA with 30 independent runs to refine the solution S, the obtained model has values $F = 4.70$, $SN = 100\%$, and $SP = 82.52\%$ on average. It reveals that OSA can effectively improve the model in terms of fitness value.

Table 11 shows the results of iTEA using artificial data with $N = 5$, 10, 15, and 30, where 3 percent, 5 percent, and 10 percent Gaussian noises were added. For the network with $N = 30$, IGA running on a single-processor PC (Pentium 4 2.8 GHz) takes 12.64 hours on average to solve the 30 subproblems with $R = 1$. The stopping condition of IGA is when the fitness value is still not effectively improved after a constant number $N_{\mathrm{gen}} = 100$ of generations. OSA performed 30 independent runs using the same initial solution S obtained from the best solutions of all individual subproblems using IGA with $R = 10$. The stopping condition of OSA is to use 3,000 iterations. The simulation results show that iTEA can effectively solve the inference problems with a value of N as large as 30. For $N = 5$ and 10, the average sensitivity (SN) performances are near or equal to 100 percent. For $N = 15$, $SN > 93\%$. For the network with $N = 30$ and 10 percent noise, $SN = 84.77\%$ and $SP = 74.58\%$. The specificity performances are ranged from 54.05 to 87.13 percent. By carefully examining the results and analyzing the inference performance, iTEA can often obtain a satisfactory S-system model, discussed below:

1. Because the Gaussian noise is added into the gene expression profiles, the original (true) values of the S-system parameters may not be the best solution to fit the noisy gene expression profiles. Note that iTEA aims to find the S-system model that can best fit the

TABLE 11
Performance of the Proposed Algorithm Where OSA Is Performed with 30 Runs

Performance	(N=5, 3%)	(N=5, 5%)	(N=10, 3%)	(N=10, 5%)	(N=15, 3%)	(N=15, 5%)	(N=30, 10%)
F(S) before OSA	1.89	2.06	1.80	6.17	3.10	7.06	140.77
Mean F(S) after OSA	1.82	1.95	1.52	4.70	3.05	7.05	115.63
Sensitivity	100%	99.13%	100%	100%	93.42%	94.74%	84.77%
(*TP*, *FN*)	(23.0, 0)	(22.8, 0.2)	(51.0, 0)	(51.0, 0)	(71.0, 5.0)	(72.0, 4.0)	(108.5, 19.5)
Specificity	54.05%	59.46%	83.43%	82.54%	86.88%	87.13%	74.58%
(*TN*, *FP*)	(20.0, 17.0)	(22.0, 15.0)	(141.0, 28.0)	(139.5, 29.5)	(351.0, 53.0)	(352.0, 52.0)	(1291.7, 440.3)
Specificity	83.51%	79.19%	92.31%	87.69%	91.09%	90.84%	84.51%
(*TN*, *FP*) Using δ_t=0.1	(30.9, 6.1)	(29.3, 7.7)	(156.0, 13.0)	(148.2, 20.8)	(368.0, 36.0)	(367.0, 37.0)	(1463.7, 268.3)

S is the combined initial solution of OSA.

noisy gene expression profiles. As a result, some small false-positive interactions may additionally occur.

2. The specificity performance highly depends on the threshold value in using the structure skeletalizing technique. When interpreting the interaction from the estimated values of the S-system parameters for noisy data, it is better to filter the small interactions using a larger threshold value. If an additional threshold $\delta_t = 0.1$ is used to filter the small interactions so that their absolute values are smaller than 0.1 by assigning a value of zero to them, the specificity performance would obviously be enhanced, ranging from 79.10 to 92.33 percent.

5 Conclusions

The S-system model has been considered suitable to characterize biochemical network systems and capable of analyzing the regulatory system dynamics. Essentially, the inference of the S-system models of genetic networks is a large-scale optimization problem consisting of $2N(N+1)$ parameters to be optimized, where N is the number of genes in the genetic network. In this paper, we propose an intelligent two-stage algorithm iTEA to search for an optimal solution to the reverse engineering problem for inference of genetic network architectures running on a single-processor PC. The algorithm iTEA solves the optimization problem using a divide-and-conquer approach in each of the two stages. At first, the original problem is decomposed into N individual $2(N+1)$-dimensional subproblems. In the first stage, an IGA is used to solve the individual subproblems independently without further estimating the gene expression levels of other genes. Our simulation demonstrated that the proposed IGA-based method is effective in solving the subproblems of inferring S-system models of genetic networks.

To compensate for the disregard of estimating accurate gene expression levels of other genes from the noisy data of gene expression, all of the solutions to the N subproblems are combined and then refined using OSA from the aspect of global optimization. In Stage 2, OSA can effectively refine the combined solution quality. From simulation results, it was shown that the proposed algorithm, iTEA, can effectively solve the inference problems with a gene number N as large as 30 from noise-free and small-noise gene expression profiles. Our future work is to parallelize iTEA to identify the dynamic pathway from real-world gene expression profiles with measurement noise, where biological knowledge is incorporated for larger networks to improve solution quality.

Acknowledgments

The authors would like to thank the National Science Council of the Republic of China for financially supporting this research under Contract NSC 95-2221-E-009-116.

References

[1] D.J. Lockhart and E.A. Winzeler, "Genomics, Gene Expression and DNA Arrays," *Nature,* vol. 405, no. 15, pp. 827-836, June 2000.

[2] P. Brazhnik, A. de la Fuente, and P. Mendes, "Gene Networks: How to Put the Function in Genomics," *Trends in Biotechnology,* vol. 20, no. 11, pp. 467-472, 2002.

[3] T. Akutsu, S. Miyano, and S. Kuhara, "Algorithms for Identifying Boolean Networks and Related Biological Networks Based on Matrix Multiplication and Fingerprint Function," *J. Computational Biology,* vol. 7, no. 3, pp. 331-343, 2000.

[4] T. Akutsu, S. Miyano, and S. Kuhura, "Identification of Genetic Networks from a Small Number of Gene Expression Patterns under the Boolean Network Model," *Proc. Pacific Symp. Biocomputing,* pp. 17-28, 1999.

[5] S. Liang, S. Fuhrman, and R. Somogyi, "REVEAL, a General Reverse Engineering Algorithm for Inference of Genetic Network Architectures," *Proc. Pacific Symp. Biocomputing,* vol. 3, pp. 18-29, 1998.

[6] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian Networks to Analyze Expression Data," *J. Computational Biology,* vol. 7, no. 3, pp. 601-620, 2000.

[7] D. Husmeier, "Sensitivity and Specificity of Inferring Genetic Regulatory Interactions from Microarray Experiments with Dynamic Bayesian Networks," *Bioinformatics,* vol. 19, pp. 2271-2282, 2003.

[8] J. Yu, V.A. Smith, P.P. Wang, A.J. Hartemink, and E.D. Jarvis, "Advances to Bayesian Network Inference for Generating Causal Networks from Observational Biological Data," *Bioinformatics,* vol. 20, pp. 3594-3603, 2004.

[9] M. Zou and S.D. Conzen, "A New Dynamic Bayesian Network (DBN) Approach for Identifying Gene Regulatory Networks from Time Course Microarray Data," *Bioinformatics,* vol. 21, no. 1, pp. 71-79, 2005.

[10] S. Kikuchi, D. Tominaga, M. Arita, K. Takahashi, and M. Tomita, "Dynamic Modeling of Genetic Networks Using Genetic Algorithm and S-System," *Bioinformatics,* vol. 19, pp. 643-650, 2003.

[11] S. Kimura, M. Hatakeyama, and A. Konagaya, "Inference of S-System Models of Genetic Networks from Noisy Time-Series Data," *Chem-Bio Informatics J.,* vol. 4, no. 1, pp. 1-14, 2004.

[12] S. Kimura, K. Ide, A. Kashihara, M. Kano, M. Hatakeyama, R. Masui, N. Nakagawa, S. Yokoyama, S. Kuramitsu, and A. Konagaya, "Inference of S-System Models of Genetic Networks Using a Cooperative Coevolutionary Algorithm," *Bioinformatics,* vol. 21, pp. 1154-1163, 2005.

[13] Y. Maki, T. Ueda, M. Okamoto, N. Uematsu, K. Inamura, K. Uchida, Y. Takahashi, and Y. Eguchi, "Inference of Genetic Network Using the Expression Profile Time Course Data of Mouse P19 Cells," *Genome Informatics,* vol. 13, pp. 382-383, 2002.

[14] R. Morishita, H. Imade, I. Ono, N. Ono, and M. Okamoto, "Finding Multiple Solutions Based on an Evolutionary Algorithm for Inference of Genetic Networks by S-System," *Proc. Congress Evolutionary Computation,* vol. 1, pp. 615-622, 2003.

[15] C. Seatzu, "A Fitting Based Method for Parameter Estimation in S-Systems," *Dynamic Systems and Applications,* vol. 9, pp. 77-98, 2000.

[16] D. Tominaga, N. Koga, and M. Okamoto, "Efficient Numerical Optimization Algorithm Based on Genetic Algorithm for Inverse Problem," *Proc. Genetic and Evolutionary Computation Conf.,* pp. 251-258, 2000.

[17] K.-Y. Tsai and F.-S. Wang, "Evolutionary Optimization with Data Collocation for Reverse Engineering of Biological Networks," *Bioinformatics,* vol. 21, pp. 1180-1188, 2005.

[18] E.O. Voit and J. Almeida, "Decoupling Dynamical Systems for Pathway Identification from Metabolic Profiles," *Bioinformatics,* vol. 20, pp. 1670-1681, 2004.

[19] A. Sorribas, S. Samitier, E.I. Canela, and M. Cascante, "Metabolic Pathway Characterization from Transient-Response Data Obtained In-Situ-Parameter-Estimation in S-System Models," *J. Theoretical Biology,* vol. 162, pp. 81-102, 1993.

[20] C.G. Moles, P. Mendes, and J.R. Banga, "Parameter Estimation in Biochemical Pathways: A Comparison of Global Optimization Methods," *Genome Research,* vol. 13, pp. 2467-2474, 2003.

[21] D. Tominaga and P. Horton, "Inference of Scale-Free Networks from Gene Expression Time Series," *J. Bioinformatics and Computational Biology,* vol. 4, no. 2, pp. 503-514, Jan. 2006.

[22] S.-Y. Ho, L.-S. Shu, and J.-H. Chen, "Intelligent Evolutionary Algorithms for Large Parameter Optimization Problems," *IEEE Trans. Evolutionary Computation,* vol. 8, no. 6, pp. 522-541, Dec. 2004.

[23] T.P. Bagchi, *Taguchi Methods Explained: Practical Steps to Robust Design.* Prentice Hall, 1993.

[24] A.S. Hedayat, N.J.A. Sloane, and J. Stufken, *Orthogonal Arrays: Theory and Applications.* Springer, 1999.

[25] S.-J. Ho, S.-Y. Ho, and L.-S. Shu, "OSA: Orthogonal Simulated Annealing Algorithm and Its Application to Designing Mixed H-2/H-Infinity Optimal Controllers," *IEEE Trans. Systems, Man, and Cybernetics-Part A: Systems and Humans,* vol. 34, no. 5, pp. 588-600, Sept. 2004.

[26] S.-J. Ho, L.-S. Shu, and S.-Y. Ho, "Optimizing Fuzzy Neural Networks for Tuning PID Controllers Using an Orthogonal Simulated Annealing Algorithm OSA," *IEEE Trans. Fuzzy Systems,* vol. 14, no. 3, pp. 421-434, June 2006.

[27] D. Thieffry, A.M. Huerta, E. Perez-Rueda, and J. Collado-Vides, "From Specific Gene Regulation to Genomic Networks: A Global Analysis of Transcriptional Regulation in Escherichia Coli," *BioEssays,* vol. 20, pp. 433-440, 1998.

[28] H. Szu and R. Hartley, "Fast Simulated Annealing," *Physics Letters,* vol. 122, pp. 157-162, 1987.

[29] S. Wei, T.W. Sun, and R.D. Wesel, "Quasi-Convexity and Optimal Binary Fusion for Distributed Detection with Identical Sensors in Generalized Gaussian Noise," *IEEE Trans. Information Theory,* vol. 47, no. 1, pp. 446-450, 2001.

Shinn-Ying Ho received the BS, MS, and PhD degrees in computer science and information engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1984, 1986, and 1992, respectively. From 1992 to 2004, he was with the Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan. He is currently a professor in the Department of Biological Science and Technology and the Institute of Bioinformatics, National Chiao Tung University, Hsinchu, Taiwan. His research interests include evolutionary algorithms, soft computing, image processing, pattern recognition, bioinformatics, data mining, machine learning, computer vision, fuzzy classifier, large-scale parameter optimization problems, and system optimization. He is a member of the IEEE.

Chih-Hung Hsieh received the BS degree in information and computer education from National Taiwan Normal University, Taipei, and the MS degree in bioinformatics from National Chiao Tung University, Hsinchu, Taiwan, in 2004 and 2006, respectively. He is currently a PhD student at the Institute of Computer Science and Information Engineering, National Taiwan University, Taipei. His research interests include evolutionary algorithms, large parameter optimization problems, system optimization, pattern recognition, and bioinformatics.

Fu-Chieh Yu received the BS degree in chemical engineering from National Chung Hsing University, Taichung, Taiwan, in 2001 and the MS degree in bioinformatics from National Chiao Tung University, Hsinchu, Taiwan, in 2006. He is currently an engineer at the Center for Measurement Standards, Industrial Technology Research Institute. His research interests include bioinformatics, evolutionary computation, and system optimization.

Hui-Ling Huang received the MS and PhD degrees in computer science and information engineering from Feng Chia University, Taichung, Taiwan, in 1998 and 2002, respectively. She is currently an assistant professor in the Department of Information Management, Jin Wen Institute of Technology, Taipei. Her research interests include evolutionary algorithms, image processing, pattern recognition, bioinformatics, data mining, machine learning, and large-scale parameter optimization problems.