Information-Guided Genetic Algorithm Approach to the Solution of MINLP Problems

Chi-Ta Young, Ying Zheng,[†] Chen-Wei Yeh, and Shi-Shang Jang*

Chemical Engineering Department, National Tsing-Huan University, Hsin Chu, Taiwan

Stochastic methods are widely used for the solution of optimization problems, because of the simplicities involved in algebraic manipulation of each particular problem. In practice, the feasible region of an engineering mixed-integer-nonlinear-programming (MINLP) problem is basically nonconvex and even rugged. Genetic algorithms (GAs) usually suffer from prematurity problems and require many different runs from different starting points, to avoid the trap of local minima. On the other hand, GAs may provide some possible local minima that have physical meanings for the engineers in its solution results. In this work, a novel genetic algorithm—called a information-guided genetic algorithm (IGA)—is developed to solve the general MINLP problems. This novel approach proposes the implementation of information theory to the mutation stage of GAs to refresh the premature population. Moreover, the detection index of prematurity of the population is based on the distances among the individuals. A local search is performed to improve the efficiency of this approach in every defined period. In this work, no initial feasible point or any problem transformation is required; thus, no additional variables and constraints are needed. On the other hand, in addition to the possible global optimum, some more local optimal solutions that may be interesting to the engineer were also found. Five examples, i.e., three multiproduct batch plant problems with different sizes, an optimization problem of regulatory metabolic reaction network, and a three-level pump network optimization problem are solved using this novel approach. The simulation results show that that the rate of convergence and discovery rate of the global minimum are substantially improved from the traditional GA.

1. Introduction

Designs of chemical and biological systems are generally mixed-integer nonlinear programming (MINLP) problems involving both discrete and continuous variables. MINLP is widely applied in many areas, from large scale designs[1] to molecular scale metabolic network design.[2] A review article[3] has shown that the application of genetic algorithms (GAs) to MINLP is very effective. The objective of this work is to derive a novel stochastic MINLP solution approach that includes the basic GA structure and an information-guided mutation stage.

The common characteristic of these MINLP problems is that these problems are nonconvex and even rugged with many local maxima/minima. The solution approaches for solving MINLP can be roughly divided into two categories:[4] deterministic approaches and stochastic approaches. Various deterministic algorithms have been published.[5,6] One of typical deterministic approach, α-BB,[7] which is a very advanced extension of Branch and Bound, overcomes this problem by implementing analytical and local smoothing techniques. A textbook by Floudas and Pardalos[8] included many of these applications, and most of them are interesting to industry. The other stream of global optimization is the stochastic algorithms, for example, simulated annealing (SA) and GAs. One of stochastic solutions to MINLP is an extension of SA.[4] On the other hand, Bjork and Nordman[9] showed that the GA is very suitable to solve a large-scale heat exchanger network. Note that the previously discussed two different approaches have their different own values; for example, a deterministic approach usually involves considerable algebra and undeviating analysis to the problem itself, whereas the evolutionary approach does not have this problem. On the other hand, an evolutionary approach basically cannot guarantee the global optimum of the problem, but some deterministic approaches, such as mathematical programming, can provide more theoretical insight of the problem.

GAs, as proposed by Holland,[10] are search algorithms based on the mechanics of natural selection and natural genetics. They combined the survival of the best fit among string structures with a structured, yet randomized, information exchange to form search algorithms with some of the innovative flair of human search. In every generation, a new set of artificial creation (strings) is created, using bits and pieces of the best fit of the old one; an occasional new point is tried for good measure. While randomized, GA is not a simple random walk. It efficiently exploits historical information to speculate on new search points with expected improved performance.

Much other attention is given to the development of GAs for MINLP. For instance, Yokota et al. developed a penalty function that is suitable for solving MINLP problems.[11] Costa and Oliveira also implemented another type of penalty function to solve various MINLP problems, including industrial-scale problems.[12] They also noted that the evolutionary approach is efficient, in terms of the number of function evaluations, and is very suitable to handle the difficulties of the nonconvexity. Going one step further, Barbosa and Lemonge[13] developed an adaptive penalty function for handling both equality and nonequality constraints. Note that the stochastic approach is based on a random sampling on the solution space. Hence, it is not guaranteed that the entire solution space is uniformly discovered. In case of highly constrained optimization, Burke et al.[14] implemented a hybrid GA to solve a time-tabling problem by combining GA with heuristic rules.

Information theory was first derived by Shannon.[15] The basic spirit of information theory is to quantify the amount of information obtained from a sampling. Hence, it is more

* To whom correspondence should be addressed. Phone: +886-3-571-3697. Fax: +886-3-571-5408. E-mail: ssjang@mx.nthu.edu.tw.
[†] Currently with Huazhong University of Science and Technology, Department of Control Science and Engineering, Wuhan, Hubei, 430074, PRC.

desirable to sample a point that possesses more information. Based on this idea, Chen et al. derived an information-driven experimental design approach by implementing the so-called information free energy and an artificial neural network meta-model to minimize the number of experiments.[16] Information theory was first implemented for a GA by Tsujimura and Gen.[17] They implemented information theory to select the best chromosome for a traveling salesman problem (TSP). Yeh and Jang[18] implemented this idea to solve some benchmark and process design problems for unconstrained nonlinear optimization problems.

In this work, we focus on the prematurity problem of GAs. A novel approach to allocate the "most informative" areas to perform efficient mutation is also derived. Furthermore, a modified local search that helps to improve the qualities of the solutions is also adopted. Five popular MINLP problems are solved to show the validity of this approach.

The remainder of this paper is organized as follows. In the next section, the problem of MINLP is stated and the background of GAs is briefly reviewed. In section 3, our approach (i.e., the information-guided genetic algorithm, IGA) is derived. In section 4, five industrial-scale examples are studied to verify our approach. In the last section, conclusive remarks are given.

2. Background

2.1. The MINLP Problem. The most general statement of a MINLP problem is

$$\min_{x,y,z} F(X,Y,Z) \tag{1}$$

$$\begin{aligned} \text{s.t.:}\quad & g_i(X,Y,Z) \geq 0 \qquad (\text{for } i = 1, ..., N_g) \\ & h_j(X,Y,Z) = 0 \qquad (\text{for } j = 1, ..., N_h) \\ & x_i^L \leq x_i \leq x_i^U \qquad (X \in R^e) \\ & y_i^L \leq y_i \leq y_i^U \qquad (Y \in I^d) \end{aligned}$$

where $F(X,Y,Z)$ is the objective function; $X = (x_1, ..., x_i, ..., x_e)$ is a vector of real variables such that for each variable x_i, there exists a upper bound x_i^U and a lower bound x_i^L; $Y = (y_1, ..., y_i, ..., y_d)$ is a vector of integer variables such that for each variable y_i, there exists a upper bound y_i^U and a lower bound y_i^L; $Z = (z_1, ..., z_i, ..., z_n)$ is a vector of binary variables; $g_i(X,Y,Z) \geq 0$ is the inequality constraint function; $h_j(X,Y,Z) = 0$ is the equality constraint function; and N_g and N_h are the number of respective constraints.

2.2. The Canonical Steps of the Genetic Algorithm. Let us consider a vector of variables as a chromosome or an individual, and the element of the vector as a gene in the GA. A set of individuals is called a population, and the population size is the number of individuals at the generation. The canonical steps of a GA can be described as follows:

Step 1. A population of candidate solutions is initialized, subject to certain constraints.

Step 2. Each chromosome in the population is evaluated through the objective function.

Step 3. New generations are generated after genetic operations such as selection, crossover, mutation, and reinsertion. *Selection* refers to selecting parents to generate offspring. *Crossover* (which is also called recombination) refers to the generation of offspring according to the characteristics of their parents. In *mutation*, traditionally, the operation typically works by selecting an individual according to the mutation rate and then the variable to be modified is chosen randomly. (In this work, it is termed "traditional mutation operation".) *Reinsertion* involves the selection of a gene group as a new generation from the offspring and parents.

Step 4. Compute the objective value and determine if the termination condition is satisfied.

Step 5. The process is halted if a suitable solution has been found or if the available computing time has expired; otherwise, the process proceeds to step 2, where the new chromosomes are scored, and the cycle is repeated.

3. The Information-Guided Genetic Algorithm (IGA)

3.1. Prematurity. Genetic operation may become trapped at a local optimum if all individuals in the population are similar. This situation is called prematurity. In conventional GAs, mutation has an important role in regard to eliminating prematurity. However, it is hard to expect that a small amount of mutated individual will still survive in the next generation while a low mutation probability is set, which means that prematurity may replay very soon. The Niching technique[19−21] is one of the very useful tools to retain the diversity of the population. Having the objective of finding all local optima, these approaches form different species, one of which is identified as a local optimum. Another helpful approach is to implement a mutation operator. Many useful mutation operators are well-known such as Flip Bit, Boundary, Non-Uniform, Uniform, and Gaussian. Recently, differential operators have been discussed and proved to be efficient mutation operators.[22,23]

In this work, prematurity is detected by calculating the difference between individuals of the same population, and information entropy is implemented to refresh the premature population efficiently.

3.2. The Information Entropy. According to Shannon's definition of information entropy,[14] for a variable V, which can randomly take a value υ from a set V, the information entropy of the set V is

$$E(V) = -\sum_{\upsilon \in \mathrm{V}} p(\upsilon) \ln p(\upsilon) \tag{2}$$

where $p(\upsilon)$ is the probability of event x occurring. If V can only take a narrow range of values, $p(\upsilon)$ for these values is ∼1. For other values of V, $p(\upsilon)$ is close to zero. Therefore, $E(V)$ is close to zero. In contrast, if V can take many different values each time with a small $p(\upsilon)$, $E(V)$ can be a large positive number. Therefore, information entropy is a measure of how random a variable is distributed.

Information entropy for a set of integers has been well-developed[24] and is easy to implement by this work. In the case of real variables, the probability and information entropy must be redefined by dividing the solution space of each variable into several subspaces. Consider the optimization problem described by expression 1, suppose the variable number is I, and $\upsilon \in \mathrm{V} = \{(V_1, V_2, ..., V_i, ..., V_I) | L_i \leq V_i \leq U_i\}$. For each variable V_i, we divide the variable V_i into R sections of equal size. Let $S = \{s_{r,i} | i = 1, ..., I, r = 1, ..., R\}$ and $s_{r,i} = [L_i^r, U_i^r]$, where

$$L_i^r = L_i + \frac{r-1}{R}(U_i - L_i) \qquad (\text{for } i = 1, ..., I \text{ and } r = 1, ..., R) \tag{3a}$$

and

$$U_i^r = U_i - \frac{R-r}{R}(U_i - L_i) \qquad \text{(for } i = 1, ..., I \text{ and } r = 1, ..., R) \quad (3b)$$

Let us define the probability that the variable V_i takes the values in subspace $s_{r,i} = [L_i^r, U_i^r]$ in the previous iterations:

$$P_{r,i} = P(V_i = v_i | v_i \in s_{r,i}) \qquad \text{(for } i = 1, ..., I \text{ and } r = 1, ..., R) \quad (4)$$

The total information entropy of the set V_i then is

$$E_i = -\sum_{r=1}^{R} P_{r,i} \log(P_{r,i}) \qquad \text{(for } i = 1, ..., I \text{ and } r = 1, ..., R) \quad (5)$$

3.3. Information-Guided Mutation Approach. Let the variable I be the number of problems, and there are J individuals in each population; the individuals then are $\{V_{i,j} | i = 1, ..., I$ and $j = 1, ..., J\}$, where $V_{i,j}$ is the ith variable of the jth individual. The detector D is computed as follows:

$$D = \sum_{i=1}^{I} \sum_{j=1}^{J} \frac{||V_{i,j} - V_{i,k}||}{M_p} \quad (6)$$

where M_p is the population size and $V_{i,k}$ is a variable of a reference individual. Let D_M be a small-valued constant.[18] If $D < D_M$, then all the individuals are similar, which means prematurity occurred. In this case, the information entropy is implemented in the mutation stage to eliminate prematurity. This novel approach is called information-entropy-guided mutation.

As the information entropy of each variable $E = \{E_1, E_2, ..., E_I\}$ is calculated, the diversity of each variable V_i ($i = 1, ..., I$) is measured. Variables with the lowest information entropy will be selected as follows:

$$E' = \{E_{\omega_1}, E_{\omega_2}, ..., E_{\omega_K} \mid E_{\omega_m} \le E_i \ (1 \le \omega_m \le I, 1 \le i \le I, \text{ and } i \ne \omega_m, \text{ where } m = 1, ..., K, \text{ for } K \le I)\} \quad (7)$$

$$\Omega = \{V_{\omega_1}, V_{\omega_1}, ..., V_{\omega_K}\} \quad (8)$$

where E' is the set of entropy parameters that have the lowest value, and Ω is the set of corresponding variables that is selected to be mutated. Note that the aforementioned idea is a novel mutation operator. For more details, the reader is referred to our previous work.[18]

In this work, the number of variables to be mutated (K) is a self-adjusted parameter, chosen as follows:

$$K = \frac{P_m I + (1 - P_m) \times I \times (D_M - D)}{D_M} \quad (9)$$

where P_m is the mutation probability. By implementing eq 9, in cases of generations with higher prematurity, more variables will be discovered.

In case $D > D_M$ (i.e., prematurity does not happen), the traditional mutation operator[11] is performed. Note that, although D_M is not a very critical parameter, a very reasonable guideline to tune this factor is to investigate the final population distance of a traditional GA and then assign their total distances as the D_M value.

3.4. Handling of Binary, Integer, and Continuous Variables. In the GA, the potential solution to each problem is denoted as a chromosome that consists of decision variables. In this work, we apply direct coding[11] for variables, which are defined in the following way: binary variables are represented in a binary code; integer and continuous variables are coded as the values within their bounds. Thus, initial chromosomes are generated randomly within variable bounds. Therefore, no feasible points are required initially in our proposed algorithm.

(1) In the selection stage, a stochastic universal sampling selection rule,[19] which can provide zero bias and minimum spread, is selected.

(2) In the recombination stage, uniform crossover (the crossover rate is 0.85) is applied to binary variables. Uniform crossover must be claimed to reduce the bias associated with the length of the binary representation used and the particular coding for a given parameter set. On the other hand, intermediate recombination[19] is adopted for integer and continuous variables. The variable values of the offspring are chosen somewhere between the variable values of the parents as follows:[19]

$$V_i = V_i^{\mathrm{Pa1}} a_i + V_i^{\mathrm{Pa2}}(1 - a_i) \qquad \text{(for } i = 1, 2, ..., I) \quad (10)$$

where a_i is a scaling factor chosen uniformly at random over an interval $[-0.25, 0.75]$ for each variable anew. The results of eq 10 for integer variables are rounded to the nearest integral values.

(3) In the mutation stage, the following approach is adopted. While the ordinary mutation operator (the mutation rate is 0.1) is activated, the binary variable values are changed from 0 to 1, or from 1 to 0; the integer variable values are changed to another random value in their corresponding bounds. While mutation is activated, mutation for binary and integer variables is determined to occur for the value with lowest subspace probability. Regardless of which mutation method is activated, for continuous variables, we choose the algorithm proposed by Mühlenbein and Schlierkamp-Voosen,[25] as follows:

$$V_i^{\mathrm{Mut}} = V_i + s_i r_i b_i \qquad \text{(for } i = 1, 2, ..., I) \quad (11)$$

where $s_i \in \{-1, +1\}$ is a uniform random number; $r_i = r \cdot$domain (where r is the mutation range (typically, $r \in [10^{-6}, 0.1]$)); $b_i = 2^{-uj}$ (where $u \in [0,1]$ is uniform at random, and $j \in \{4, 5, ..., 20\}$ is the mutation precision.

(4) In the reinsertion stage, the local search for the individuals after the mutation stage is first performed (see section 3.6). Elitist reinsertion is then implemented to choose the best chromosomes from the parents and offspring, and the number of the chromosomes is selected and set to meet the population size. Note that the chromosomes obtained from the information-guided variables are kept in the new generation.

3.5. Handling of Equality and Inequality Constraints. Most engineering problems are constrained optimization problems. Penalty function approaches are often used to solve these problems. A penalty term is added to the objective function. Using this modification, both feasible solutions and infeasible solutions can be evaluated in the optimization procedure. The original constrained optimization problems are converted to unconstrained optimization problems. Although the penalty function requires different parameters in terms of different problems, it is used widely, because of its simplicity and convenience.

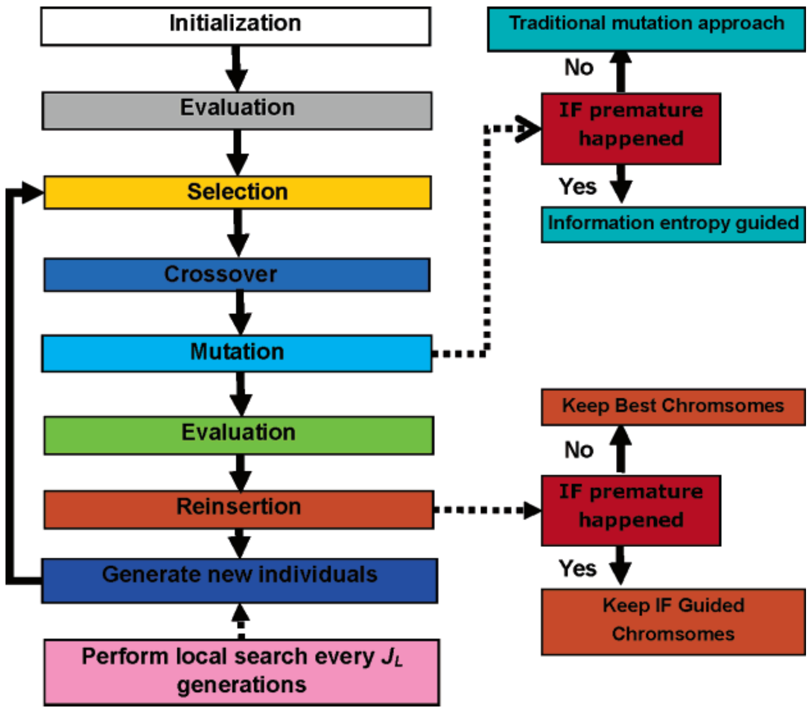

Figure 1. Flowchart of IGA.

Table 1. Possible Global Optimum and Two Local Optimal Solutions

	N	*S*	*B*	T_L	*C*
possible global optimum	1 1 1	480 720 960	240 120	20 16	38499.8
local optimums					
No. 1	2 2 1	250 360 480	120 60	10 8	40977
No. 2	2 1 1	373.33 560 746.67	186.67 93.333	20 8	41844

In this work, the penalty function[26] is extended to examine continuous variables, binary variables, and integer variables as follows. Set

$$F(V) = \begin{cases} f(V) & \text{(if } V \text{ is feasible)} \\ \bar{f}(V) + \sum_{j=1}^{m} k_j vo_j(V) & \text{(if otherwise)} \end{cases} \tag{12}$$

where

$$\bar{f}(V) = \begin{cases} f(V) & \text{(if } f(V) > \langle f(V)\rangle) \\ \langle f(V)\rangle & \text{(if otherwise)} \end{cases}$$

$$k_j = |\langle f(V)\rangle| \frac{\langle vo_j(V)\rangle}{\sum_{l}^{m} [\langle vo_l(V)\rangle]^2}$$

$$vo_j(VR) = \begin{cases} |h_j(V)| & \text{(for an equality constraint)} \\ \max\{0, -g_j(V)\} & \text{(otherwise)} \end{cases}$$

$\langle f(V)\rangle$ is the average objective function values of the current generation, $vo_j(V)$ ($j = 1, ..., m$) is the amount of violation of the jth constraint by the candidate solution V, $\langle vo_l(V)\rangle$ is the violation of the lth constraint averaged over the current population, m denotes the number of constraints to be penalized, h represents the equality constraints, and g represents the inequality constraints.

This penalty function does not require any predefined parameters except the values of the average objective function and the constraints' violations. It is a simple, adaptive, parameterless penalty scheme for the solution of constrained problems via GAs.

3.6. Local Search. Local search can improve the GA solutions. Ombuki and Ventresca[27] mentioned that local search during mutation will accelerate at a convergent speed but extend the computation time. In their case, the problem was an unconstrained discrete time system.

In the case of constrained optimization, the stochastic approach adopted in this work has a major concern in regard to feasibility of the new generation. In this work, the following local searches over a period of a certain number of generations are performed before the stage of reinsertion, using the reinserted points as initial points. By fixing the integer variable Y and the binary variable Z, the following problem is solved by a general-purpose nonlinear programming (NLP) solver, such as *fmincon* in MATLAB:

$$\min_{x} \quad F(X,Y,Z) \tag{13}$$

$$\begin{aligned} \text{s.t.:} \quad & g_i(X,Y,Z) \geq 0 \qquad (\text{for } i = 1, ..., N_g) \\ & h_j(X,Y,Z) = 0 \qquad (\text{for } j = 1, ..., N_h) \\ & x_i^{\mathrm{L}} \leq x_i \leq x_i^{\mathrm{U}} \qquad (X \in R^{\mathrm{e}}) \end{aligned}$$

Although it may bring a small increase in the function calls, it will improve the feasibility (section 4.3) and the quality of the solutions greatly (see sections 4.1 and 4.2).

3.7. The Algorithm. The algorithm is shown in Figure 1, which follow the canonical steps of a GA described in section 2.2. In the mutation stage, the distance test described in section 3.3 is performed first. In the case of prematurity, information-guided mutation is performed and these chromosomes are kept together with the best chromosomes among the remainders in the reinsertion stage. Otherwise, the traditional mutation and

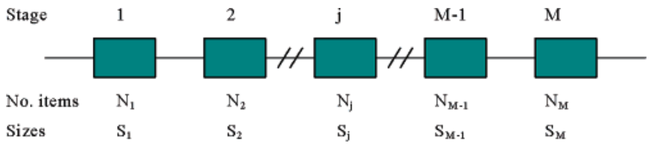

Figure 2. Multiproduct batch problem.

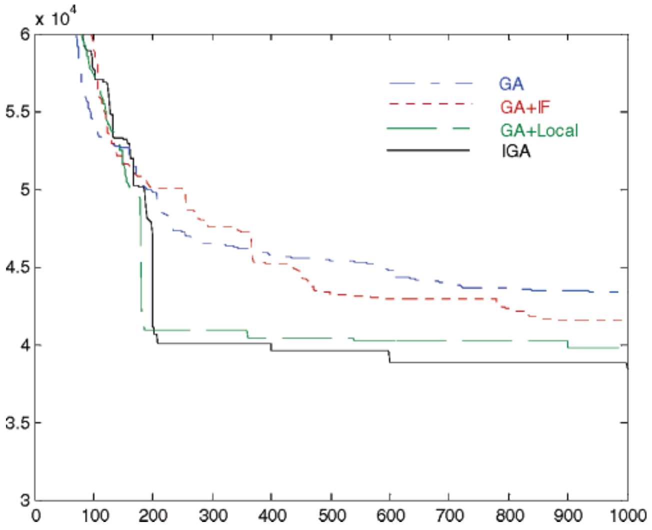

Figure 3. Comparisons of IGA with other GAs for multiproduct batch example ($M = 3$, $N = 2$).

Table 2. Comparison of the Statistics of Four Different GAs for the Multiproduct Batch Example ($M = 3$, $N = 2$)

	GA	GA+IF	GA+local	IGA (GA+IF+local)
average	43370	41502	39799	38499
standard deviation	1901.8	2577.1	1708.5	7.6695×10^{-12}
number of function evaluations	25905	25198	28598	30867

reinsertion approaches are performed. Furthermore, a local search in every J_L generations, to find more-feasible solutions, is implemented as shown in Figure 1.

The termination criterion is used to stop the optimization process. In our cases, when a given maximum number of generations is reached, the search process is stopped and the best solutions are returned.

4. Case Studies

Numerical experiments are conducted to show the validity of our theory developed in the previous section. Five industrial scaled problems are solved. In section 4.1, three multiproduct batch plant problems of different sizes are solved. Compared with the original reports, the frequency of discoveries of the global minimum is highly enhanced in the first problem and better optimum and alternative solutions are found in the second example. In section 4.2, the optimization of regulatory metabolic reaction network is solved without further variable transformations. Other local optimal solutions that are of interest of design engineers are provided by this approach. In section 4.3, a three-level pump network optimization problem is solved without variable transformations.

4.1. Multiproduct Batch Plant Problem. The following multiproduct batch problem was initially studied by Grossmann and Sargent.[5] As shown in Figure 2, N different products are manufactured on M batch processing stages. In each stage j, N_j units operate independently and all the units have the same size S_j; the time required to process one batch of product i in stage j is T_{ij}. For each product, B_i is the size, n_i the number of the batches, Q_i the total production, T_{Li} the maximum cycling time. The design problem is to choose N_j, S_j, B_i, and T_{Li} to minimize the capital cost C of the plant. It becomes the following MINLP problem:

$$\min C = \min \sum_{j=1}^{M} \alpha_j N_j S_j^{\beta_j}$$

$$\text{s.t.} \sum_{i=1}^{N} \frac{Q_i T_{Li}}{B_i} \le H$$

$$S_j \ge S_{ij} B_i \qquad (\text{for } i = 1, 2, ..., N, j = 1, 2, ..., M)$$

$$N_j T_{Li} \ge t_{ij} \qquad (\text{for } i = 1, 2, ..., N, j = 1, 2, ..., M)$$

$$1 \le N_j \le N_j^u \qquad (\text{for } j = 1, 2, ..., M)$$

$$S_j^l \le S_j \le S_j^u \qquad (\text{for } j = 1, 2, ..., M)$$

$$T_{Li}^l \le T_{Li} \le T_{Li}^u \qquad (\text{for } i = 1, 2, ..., N)$$

$$B_i^l \le B_i \le B_i^u \qquad (\text{for } i = 1, 2, ..., N)$$

$$N_j \equiv \text{integer}$$

$$T_{Li}^l = \max \frac{t_{ij}}{N_j^u}$$

$$T_{Li}^l = \max t_{ij}$$

$$B_i^l = \frac{Q_i}{H} T_{Li}$$

$$B_i^u = \min\left(Q_i, \min \frac{S_j^u}{S_{ij}}\right) \qquad (14)$$

where H is the given period of time, α_j and β_j are appropriate cost parameters, $S_{ij} > 0$, $t_{ij} > 0$ are constants, and the terms N_j^u, S_j^l, and S_j^u are the given limits.

(1) $M = 3$, $N = 2$. This case has been studied in many previous works (e.g., Kocis and Grossman,[6] Angira and Babu[23]). In this example, the following values are implemented for the IGA, as depicted in section 3.7: $D_M = 50$, $J_L = 200$, maximum number of generations $= 1000$, and $M_p = 14$. Without further transformation of the problem, the same global optimum as that provided in previous works is obtained, as listed in Table 1. Note that two different local optima, which may be of interest to the design engineer, are also provided by this algorithm, as listed in Table 1.

Figure 3 gives the histories of the decreasing of the objective function, as a function of number of generations by IGA. Note that the plot is based on the averages of 20 simulation runs. To show the necessity of the modification of the traditional GA, IGA is compared with traditional GA (no IF, no local search), GA plus information-guided mutation (GA+IF), and GA plus local search only (GA+local search). It is obvious that only the approach proposed in this work converges to a possible global minimum with a satisfactory speed.

The comparison of the statistical properties of the "best" solutions found by the four methods is shown in Table 2. Compared with other GA methods, IGA yields the lowest average objective function values and much lower standard deviations (STDs). Furthermore, the average number of function calls per run using our approach is 30 867, which is much lower than the data reported by a modified GA-based original work.[23]

Table 3. Possible Global Optimum and Two Local Optimal Solutions

	N	S	B	T_L	C
possible global optimum	2 2 3 2 1 1	3000 1891.6 1974.7 2619.1 2328.1 2109.8	379.75 770.31 727.52 638.3 525.43	3.2 3.4 6.2 3.4 3.7	285 510
local optimum	2 2 3 2 2 1	2898.8 2143.3 1939.6 2831.4 2494.7 2324.1	354.99 816.83 771.34 552.61 509.62	3.2153 3.633 6.4586 3.4808 2.2286	319 360

Table 4. Comparison of the Statistics of Four Different GAs for the Multiproduct Batch Example (M = 6, N = 5)

	GA	GA+IF	GA+local	IGA (GA+IF+local)
average	299140	293150	292960	287200
standard deviation	24156	5033.5	22868	7570.3
number of function evaluations	51843	51809	78089	73075

(2) $M = 6$, $N = 5$. This problem was studied by Kocis and Grossman.[6] Originally, 22 variables and 61 inequality constraints existed in this problem. The problem size is equivalent to solving 4096 NLP subproblems. Cardosa et al.[4] implemented a SA approach with some modifications to the original problem to solve this problem, and they reported the optimal and suboptimal solutions that are listed in Table 2.[4] They claimed that, although the rate of discovery of the ill-conditioned possible global optimum is only 2.6%, they only implemented half of the constraints given by the original work,[6] because of the neglect of variable transformations. Hence, additional algebraic efforts become unnecessary.

In this example, the following values are implemented for the IGA, as depicted in section 3.7: $D_M = 50$, $J_L = 400$, the maximum number of generations = 2000, and $M_p = 14$. Without further transformation of the problem, 22 variables and only 61 constraints, the possible global optimum, which is consistent with the original work by Kocis and Grossman,[6] and one local minimum, are also obtained, as shown in Table 3.

Figure 4 gives the history of the decreasing of the objective function, as a function of the number of generations by IGA. Note that the plot is based on the averages of 20 simulation runs. To show the necessity of the modification of the traditional GA, IGA is compared with traditional GA (no IF, no local search), GA+IF, and GA+local search. It is obvious that only the approach proposed in this work converges to a global minimum with a satisfactory speed.

The comparison of the statistical properties of the "best" solutions found by the four methods based on 20 runs is shown in Table 4. The four different GAs terminate in case the best fitness value of each generation does not change within the last 10 iterations. Compared to other GA methods, IGA yields the lowest average objective function values, one-third or lower STDs, and the highest frequency of occurrence of possible global optimum. Furthermore, the average number of function calls per run using our approach is 73 075, which is only 8.8% of the original report (831 149 function calls) by Cardosa et al.;[4] and the occurrence of possible global optimum is 95%, compared with the value of 2.6% that has been given by Cardosa et al.[4]

(3) $M = 12$, $N = 8$. An extension of the aforementioned problem to $M = 12$, $N = 8$ is studied in this section. By implementing mathematical programming, 40 variables and 217 constraints should be implemented. The problem size is equivalent to solving 2.4414×10^8 NLP subproblems (see, e.g., Goyal and Ierapetritou[28]). The problem parameters are set such that $V_j^L = 1000l$, $V_j^U = 10000l$. For other parameters, the reader is referred to Appendix B of the original work.[28]

In this work, the following values are implemented for the IGA, as depicted in section 3.7: D_M =300, J_L =300, the maximum number of generations = 2600, and $M_p = 14$. Without further transformation of the problem, 40 variables and only 193 constraints, which is fewer than the original report,[28] are implemented. The possible global optimum and three nearby local optimal solutions that have been discovered by IGA are listed in Table 5.

Again, Figure 5 gives the histories of the decreasing of the objective function, as a function of the number of generations,

Table 5. Possible Global Optimum and Two Local Minima Discovered by IGA

	N	S	B	T_L	C
possible global optimum	2 2 2 2 2 1 1 1 1 2 2 1	10000 5949.4 6582.3 6219.5 7721.5 5319.1 6036.5 5674.8 7649.6 7150.7 6984.4 7649.6	1265.8 1829.3 1662.9 2127.7 1382.1 1091.3 1265.8 2417.8	4.15 3.4 6.2 3.4 2.85 2.5 3.2 2.15	1.903×10^6
		Local Optimums			
No. 1	2 2 3 2 2 1 1 1 1 2 1 1	10000 5949.4 6582.3 6909.3 7721.5 5627.5 6706.1 6096.4 8522.7 7966.9 7781.6 8522.7	1265.8 2032.1 1852.8 2127.7 1535.4 1172.4 1265.8 2557.9	3.8 3.4 6.2 3.4 4.2 2.7 3.2 3.4	1.934×10^6
No. 2	3 2 3 2 2 1 1 1 1 2 2 1	9885.7 5881.4 6507.1 6131.7 7633.3 5258.4 5882 5347.3 7208.1 6738 6581.3 7208.1	1251.4 1782.4 1567 2103.4 1362.6 1028.3 1251.4 2390.2	3.8 3.2 6.2 3.4 2.85 2.5 3.2 2.15	1.979×10^6
No. 3	2 2 3 2 3 2 1 1 1 3 2 2	7960.3 4735.8 5239.7 4937.4 6146.5 4232 4792.2 4356.5 5872.5 5489.5 5361.9 5872.5	1007.6 1452.2 1276.6 1692.8 1097.2 837.79 1007.6 1923.7	3.2 3.4 5.95 2.2 2.1 1.4167 2.1 2.15	2.036×10^6

Table 6. Comparison of the Statistics of Four Different GAs for the Multiproduct Batch Example (M = 12, N = 8)

	GA	GA+IF	GA+local	IGA (GA+IF+local)
average	2.1609×10^6	2.1773×10^6	2.0822×10^6	1.9648×10^6
standard deviation	81577	96379	1480700	47598
number of function evaluations	41463	41428	63098	91984

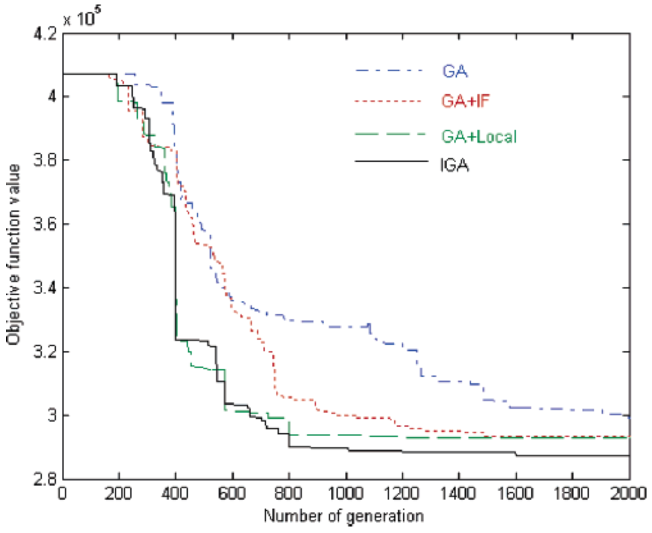

Figure 4. Comparisons of IGA with other GAs for multiproduct batch example ($M = 6$, $N = 5$).

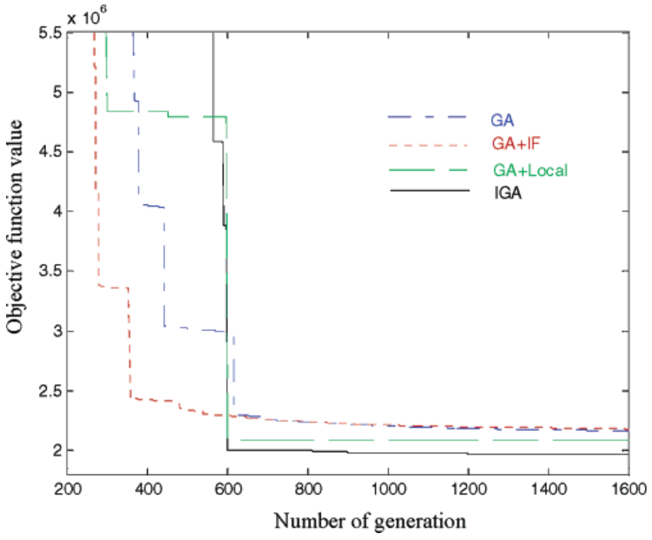

Figure 5. Comparisons of IGA with other GAs for multiproduct batch example ($M = 12$, $N = 8$).

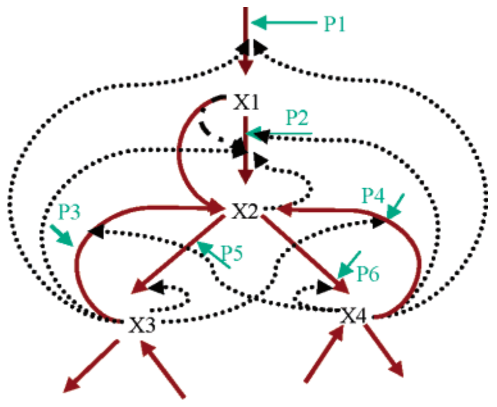

Figure 6. Superstructure of the XMP and GMP synthesis pathway.

based on the average of 20 simulation runs. The four curves in Figure 5 again represent the cost at a certain generation using the method with or without information entropy and local search. Once again, only the approach proposed by this work discovers the global minimum with a satisfactory speed. The statistical comparison of the four methods based on 20 runs is shown in Table 6. The four different GAs are terminated in case the best fitness value of each generation does not change within the last 10 iterations. Compared to other GA methods, although IGA requires a slightly higher number of average function calls per run, it has the lowest average and second-lowest STD of the objective function values.

4.2. The Regulatory Metabolic Reaction Network Problem. In this section, the same example that was studied by Hatzimanikatis et al.[2] is studied. It involves yield optimization in xanthine monophosphate (XMP) and guanosine monophosphate (GMP) production. As shown in Figure 6, X_j represents the concentration of the metabolite j ($j = 1, ..., 4$), P_l represents the amount of manipulated variable l ($l = 1, ..., 6$), the dashed lines denote inhibition, and dashed−dotted lines denote activation. The S-system representation allows for the description of biochemical systems by nonlinear models of power-law form. The S-system representation of the aforementioned pathway is

$$\frac{\mathbf{d}\boldsymbol{X}_1}{\mathbf{d}\boldsymbol{t}} = 900\boldsymbol{X}_3^{-0.5}\boldsymbol{X}_4^{-0.5}\boldsymbol{P}_1 - 10\boldsymbol{X}_1^{0.5}\boldsymbol{X}_2^{-0.1}\boldsymbol{X}_3^{-0.2}\boldsymbol{X}_4^{-0.2}\boldsymbol{P}_2^{0.6}\boldsymbol{P}_3^{0.4}$$

$$\frac{\mathbf{d}\boldsymbol{X}_2}{\mathbf{d}\boldsymbol{t}} = 7.34\boldsymbol{X}_1^{0.308}\boldsymbol{X}_2^{-0.062}\boldsymbol{X}_3^{-0.162}\boldsymbol{X}_4^{-0.1}\boldsymbol{P}_2^{0.37}\boldsymbol{P}_3^{0.245}\boldsymbol{P}_4^{0.385} - 43.8\boldsymbol{X}_2^{0.42}\boldsymbol{X}_3^{-0.339}\boldsymbol{X}_4^{-0.5}\boldsymbol{P}_5^{0.4}\boldsymbol{P}_6^{0.6}$$

$$\frac{\mathbf{d}\boldsymbol{X}_3}{\mathbf{d}\boldsymbol{t}} = 2.71\boldsymbol{X}_2^{0.409}\boldsymbol{X}_3^{-0.387}\boldsymbol{P}_5^{0.455} - 0.036\boldsymbol{X}_1^{0.041}\boldsymbol{X}_3^{0.43}\boldsymbol{X}_4^{-0.014}\boldsymbol{P}_3^{0.28}$$

$$\frac{\mathbf{d}\boldsymbol{X}_4}{\mathbf{d}\boldsymbol{t}} = 13.03\boldsymbol{X}_2^{0.041}\boldsymbol{X}_4^{-0.399}\boldsymbol{P}_6^{0.405} - 0.143\boldsymbol{X}_3^{-0.026}\boldsymbol{X}_4^{0.40}\boldsymbol{P}_4^{0.26}$$

Setting $\boldsymbol{Y}_{\mathrm{i}} = \ln \boldsymbol{X}_{\mathrm{i}}$, the optimization problem becomes

$$\boldsymbol{P}_{\mathrm{i}} = 1 \qquad (\text{for } i = 1, ..., 6) \tag{15}$$

$$\max\ (\mathrm{y4})$$

$$\text{s.t. } -0.5y_1 + 0.1y_2 - 0.3y_3 - 0.3y_4 - z_{13}\epsilon_{13}y_3 - z_{14}\epsilon_{14}y_4 + 0.6z_{21}\epsilon_{21}y_1 + 0.6z_{22}\epsilon_{22}y_2 + 0.6z_{23}\epsilon_{23}y_3 + 0.6z_{24}\epsilon_{24}y_4 + 0.4z_{34}\epsilon_{34}y_4 + w_1q_1 - 0.6w_2q_2 - 0.4w_3q_3 = -4.4998$$

$$0.308y_1 - 0.482y_2 + 0.177y_3 + 0.4y_4 - 0.37z_{21}\epsilon_{21}y_1 - 0.37z_{22}\epsilon_{22}y_2 - 0.37z_{23}\epsilon_{23}y_3 - 0.37z_{24}\epsilon_{24}y_4 - 0.245z_{34}\epsilon_{34}y_4 - 0.385z_{43}\epsilon_{43}y_3 + 0.4z_{53}\epsilon_{53}y_3 + 0.6z_{64}\epsilon_{64}y_4 + 0.37w_2q_2 + 0.245w_3q_3 + 0.385w_4q_4 - 0.4w_5q_5 - 0.6w_6q_6 = 1.7863$$

$$-0.14y_1 + 0.409y_2 - 0.817y_3 - 0.014y_4 - 0.455z_{53}\epsilon_{53}y_3 + 0.287z_{34}\epsilon_{34}y_4 - 0.28w_3q_3 + 0.455w_5q_5 = -4.3212$$

$$0.041y_2 + 0.026y_3 - 0.799y_4 - 0.405z_{64}\epsilon_{64}y_4 + 0.26z_{43}\epsilon_{43}y_3 - 0.26w_4q_4 + 0.405w_6q_6 = -4.5122$$

$$\text{bounds on } y_j \text{ (for } j = 1, 2, 3\text{):} \quad \begin{cases} \ln(4.9) \geq y_1 \geq \ln(6.0) \\ \ln(192) \geq y_2 \geq \ln(234) \\ \ln(2176) \geq y_3 \geq \ln(2660) \end{cases}$$

$$\text{bounds on } P_l \text{ (for } l = 1, ..., 6\text{):} \quad \ln(P_l^{\mathrm{L}}) \geq P_l \geq \ln(P_l^{\mathrm{U}}) \tag{16}$$

where $\boldsymbol{w}_{\mathrm{l}}$ ($l = 1, ..., 6$) and {$\boldsymbol{z}_{13}$, $\boldsymbol{z}_{14}$, $\boldsymbol{z}_{21}$, $\boldsymbol{z}_{22}$, $\boldsymbol{z}_{23}$, $\boldsymbol{z}_{24}$, $\boldsymbol{z}_{34}$, $\boldsymbol{z}_{43}$, $\boldsymbol{z}_{53}$, $\boldsymbol{z}_{64}$} are binary variables, $\boldsymbol{y}_{\mathrm{j}}$ ($j = 1, ..., 4$) and $\boldsymbol{q}_{\mathrm{l}}$ ($l = 1, ..., 6$) are continuous variables, L represents the lower bound, and U represents the upper bound.

Table 7. Values of Variables in the Original Solution and IGA Solutions 1−5

	w (w_1−w_6)	z(z_{13}, z_{14}, z_{21}, z_{22}, z_{23}, z_{24}, z_{34}, z_{43}, z_{53}, z_{64})	q (q_1−q_6)	x_4
original solution	1 1 1 1 1 1	0 1 0 1 1 1 0 0 0 1	1.6094 1.6094 −0.3634 1.6094 −0.4354 1.2513	55015.6
IGA solution 1	1 1 0 1 0 0	0 1 0 1 1 1 0 0 0 1	1.4722 1.2906 −1.3646 0.42012 0.17209 −0.29841	52575
IGA solution 2	1 1 1 0 1 0	0 1 0 1 1 1 1 0 0 1	1.2309 1.6094 −1.6094 0.60501 −0.23505 −0.17344	65121
IGA solution 3	1 1 0 0 0 1	0 1 1 1 1 1 0 1 0 1	1.6094 1.6094 1.0534 1.5246 1.6027 0.29228	56613
IGA solution 4	1 1 1 0 1 1	0 1 0 1 1 1 0 1 0 1	1.534 1.6072 −0.45455 0.77799 −0.20221 0.65065	69633
IGA solution 5	1 1 0 1 0 0	0 1 0 0 1 1 0 0 0 1	1.126 1.4727 −0.1548 0.4643 0.0361 −1.2034	49811

In this example, the following values are implemented for IGA: $D_M = 1$, $J_L = 400$, the maximum number of generations = 2600, and $M_p = 14$. After 2600 generations, several local minima with different network structures are found. As discussed in the work of Hatzimanikatis et al.,[2] it is very important to find a feasible solution with the least modification from the original structure of the network. Without further transformation of the original problem of eq 15, using 26 variables (compared to 42 variables of the original work) and 13 constraints (compared to 77 constraints of the original work), 5 local optima are discovered and are listed in Table 7.

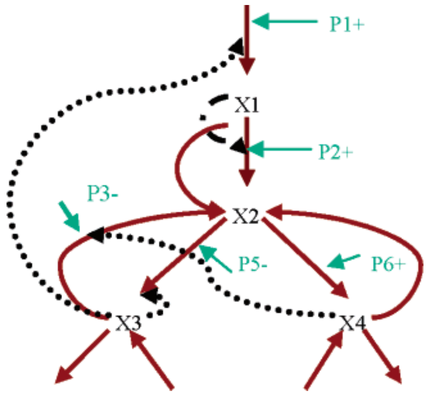

Figure 7. Network structure of solution 4.

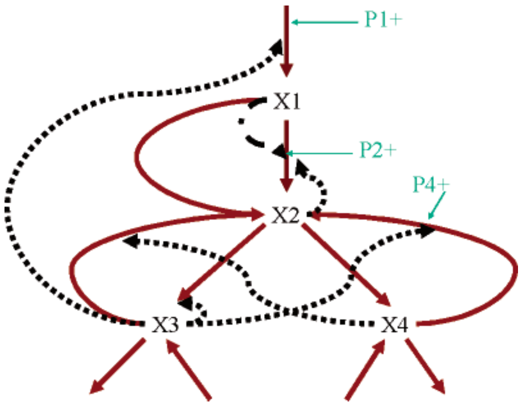

Figure 8. Network structure of solution 5.

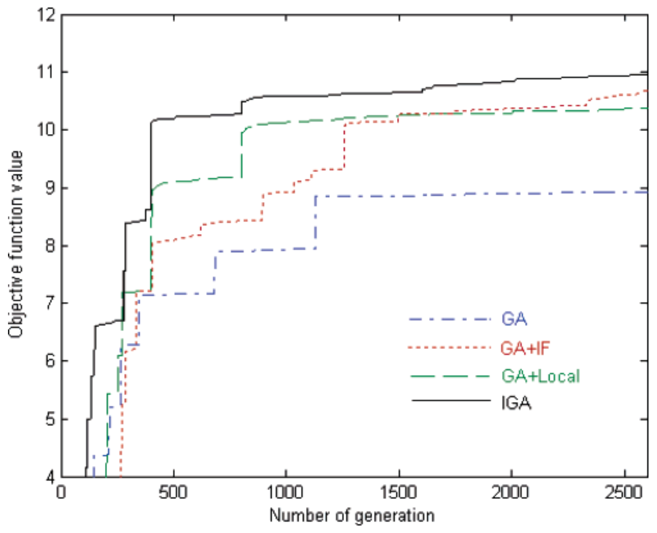

Figure 9. Comparisons of IGA with other GAs for the regulatory metabolic reaction network example.

Table 8. Comparison of the Statistics of Four Different GAs for the Regulatory Metabolic Reaction Network Example

	GA	GA+IF	GA+local	IGA (GA+IF+local)
average	8.9284	10.684	10.375	10.951
standard deviation	3.4368	0.46561	1.3125	0.32403
number of function evaluations	67359	67358	79594	78670

Figures 7 and 8 show network structures of solutions 4 and 5, which are listed in Table 7. The values of the variables $\mathbf{w}_l$ (l = 1, ..., 6) ,{$\mathbf{z}_{13}$, $\mathbf{z}_{14}$, $\mathbf{z}_{21}$, $\mathbf{z}_{22}$, $\mathbf{z}_{23}$, $\mathbf{z}_{24}$, $\mathbf{z}_{34}$, $\mathbf{z}_{43}$, $\mathbf{z}_{53}$, $\mathbf{z}_{64}$}, $\boldsymbol{q}_l$ (l = 1, ..., 6), and x_4 in each solution are shown in Table 7. Compared to the original yield reported by the work (i.e., $x_4 = 55015.6$), it is clear that solution 4 gives 26.6% higher yields. However, solution 4 requires one more modification on the metabolic network, as shown in Figure 7, although less modifications on the enzymes are required. Furthermore, some network structures, such as solution 5 suggested by this work, have less modification from the original metabolic structure, and this is of the interest of design engineers, as shown Figure 8, although slighty less product yield can be achieved, as shown in Table 7. Note that each "1" of w and z in Table 7 indicates a modification to the original structure. As shown in Table 7, most solutions obtained uaing this approach have less modification to the original structure.

Again, Figure 9 gives the histories of the decreasing of the objective function, as a function of the number of generations, based on the average of 50 simulation runs. As shown in Figure 9, the performance of IGA is superior to other GAs. The comparison of the statistics of all four approaches is shown in Table 8. Compared to other GA methods, although the proposed method yields the higher average number of function calls per run, the statistics of the objective function values are much superior to other GAs.

4.3. Pump Network Problem. A schematic plot of a three-level pump network configuration is given in this section. As described by Westerlund et al.,[29] the problem can be summarized as follows. Select the best pump or configuration of pumps coupled in series and/or parallel. Given the pressure rise (total head), as a function of the capacity of a set of centrifugal pumps, as well as the total required pressure rise and total flow for the configuration to be selected, the MINLP problem can be expressed as follows:

$$\min_{x,\upsilon,\omega,P,\Delta P,N_p,N_s,z} \sum_{i=1}^{3} (C_i + C'_i P_i) N_{p_i} N_{s_i} z_i$$

$$\text{s.t. } P_1 - 19.9\left(\frac{\omega_1}{\omega_{\max}}\right)^3 - 0.1610\left(\frac{\omega_1}{\omega_{\max}}\right)^2 \upsilon_1 + 0.000561\left(\frac{\omega_1}{\omega_{\max}}\right)\upsilon_1^2 = 0$$

$$P_2 - 1.21\left(\frac{\omega_2}{\omega_{\max}}\right)^3 - 0.0644\left(\frac{\omega_2}{\omega_{\max}}\right)^2 \upsilon_2 + 0.000564\left(\frac{\omega_2}{\omega_{\max}}\right)\upsilon_2^2 = 0$$

$$P_3 - 6.52\left(\frac{\omega_3}{\omega_{\max}}\right)^3 - 0.1020\left(\frac{\omega_3}{\omega_{\max}}\right)^2 \upsilon_3 + 0.000232\left(\frac{\omega_3}{\omega_{\max}}\right)\upsilon_3^2 = 0$$

$$\Delta p_1 - 629\left(\frac{\omega_1}{\omega_{\max}}\right)^2 - 0.696\left(\frac{\omega_1}{\omega_{\max}}\right)\upsilon_1 + 0.0116\upsilon_1^2 = 0$$

$$\Delta p_2 - 215\left(\frac{\omega_2}{\omega_{\max}}\right)^2 - 2.950\left(\frac{\omega_2}{\omega_{\max}}\right)\upsilon_2 + 0.0115\upsilon_2^2 = 0$$

$$\Delta p_3 - 361\left(\frac{\omega_3}{\omega_{\max}}\right)^2 - 0.530\left(\frac{\omega_3}{\omega_{\max}}\right)\upsilon_3 + 0.00946\upsilon_3^2 = 0$$

$$x_1 + x_2 + x_3 = 1$$

$$\upsilon_i N_{p_i} - x_i V_{tot} = 0 \qquad (\text{for } i = 1, 2, 3)$$

$$\Delta P_{tot} z_i - \Delta p_i N_{s_i} = 0 \qquad (\text{for } i = 1, 2, 3)$$

$$\omega_i - \omega_{\max} z_i \leq 0 \qquad (\text{for } i = 1, 2, 3)$$

$$P_i - P_{\max,i} z_i \leq 0 \qquad (\text{for } i = 1, 2, 3)$$

$$\Delta p_i - \Delta P_{tot} z_i \leq 0 \qquad (\text{for } i = 1, 2, 3)$$

$$\upsilon_i - V_{tot} z_i \leq 0 \qquad (\text{for } i = 1, 2, 3)$$

$$x_i - z_i \leq 0 \qquad (\text{for } i = 1, 2, 3) \tag{17}$$

where z_i represents binary variables, which denote the existence of level i; $N_{pi} \in \{0,1,2,3\}$ and $N_{si} \in \{0,1,2,3\}$ are the integer variables, where N_{p_i} denotes the number of parallel lines and N_{s_i} denotes the number of pumps in series at level i; $x_i \subset [0,1]$, $\upsilon_i \subset [0,V_{tot}]$, $\omega_i \subset [0,\omega_{\max}]$, $P_i \subset [0,P_{\max,i}]$ and $\Delta p_i \subset [0,\Delta P_{tot}]$ are continuous variables, where x_i is the fraction of total flow going to level i, υ_i the flow rate on each line, ω_i the rotation speed of all pumps, P_i the power requirement, and Δp_i the pressure increase at level i. The details of the problem can be found in section 12.5 of Floudas et al.[30] Note that no variable transformation is implemented in this work; hence, no additional variables or constraints are added. Algebraic operation is significantly reduced.

In this example, the following values are implemented in the proposed IGA: $D_M = 100$, $J_L = 150$, the maximum number of generations = 1200, and $M_p = 14$. The possible global optimum with the objective function value 128 894 FIM/yr is obtained, corresponding to $z = [1\ 1\ 0]$, $Np = [2\ 1\ 0]$, $Ns = [1\ 2\ 0]$, $x = [0.91429\ 0.085714\ 0]$, $\omega = [2855.1\ 2950\ 0]$, $P = [28.27\ 2.63\ 0]$, and $\Delta p = [400\ 200\ 0]$.

Figure 10 shows the objective values, as a function of the number of generations, using two different types of GAs, based on the averages of 50 runs. Unlike the previous cases, traditional GA and GA+IF approaches failed to discover the global minimum, because of the importance of the local search to guarantee the feasibility of the new generation, as mentioned at section 3.6. These two cases are not included to compare the performance of different GAs. As shown in Figure 10, again, the performance of IGA is superior to the GA+local search case. Table 9 reveals that IGA is superior to the GA+local search in the average and STD of the solution of 50 different runs.

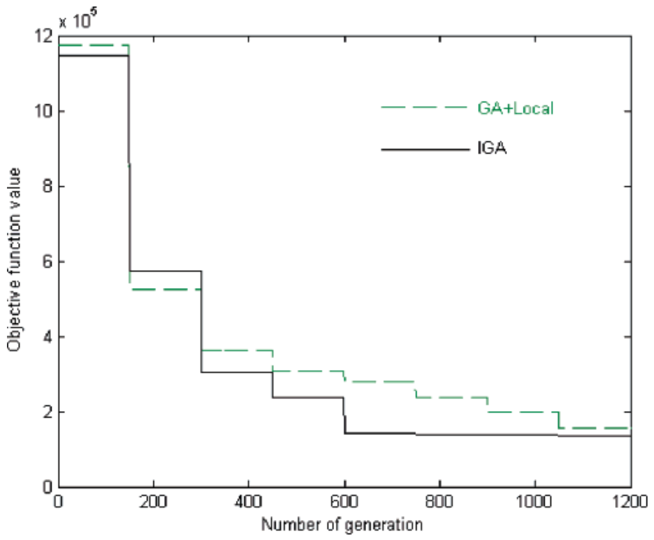

Figure 10. Comparisons of the performance of IGA with GA+local search.

Table 9. Comparison of the Statistics of Two Different GAs for the Pump Network Problem

	GA+local search	IGA
average	156470	135890
standard deviation	15758	4992.5
number of function evaluations	55374	47958

5. Conclusion

The most general formulation of a chemical process design is essentially the optimization of a mixed-integer nonlinear programming (MINLP) problem. In this paper, a novel genetic algorithm (IGA) that substantially improves the efficiency of the traditional genetic algorithm (GA) is derived. This approach implements the information theory to refresh the population as prematurity occurs. A modified local search is performed to determine the more-feasible solutions in a constant period of generations. Five popular numerical examples are solved using this approach. In this work, the proposed approach is easier to implement without additional algebraic effort than some mathematical programming approaches are. Besides, this approach provides physically meaningful local optima for the consideration of the design engineer. By comparing the results of five examples, IGA demonstrates superior performance to the existing stochastic MINLP approaches. The simulation results also showed that this novel optimizer is valid and efficient for real applications.

Nomenclature

a_i	= a scaling factor
B_i	= the size of product i (for i = 1, ..., N)
C	= the capital cost of the plant
d	= number of integer variables

D_M = a constant set as a threshold
e = number of continuous variables
E = information entropy of a set
E_i = information entropy of a variable V_i (for $i = 1, ..., I$)
E' = a set of lower entropy
F = objective function
g_i = inequality constraint (for $i = 1, ..., N_g$)
h_j = equality constraint (for $j = 1, ..., N_h$)
H = the given period of time
I = number of variables
J = number of individuals in each population
J_L = the number of generations to perform each local search
k_j = penalty parameter (for $j = 1, ..., m$)
K = number of mutation variables
L_i = lower bound of variables V_i (for $i = 1, ..., I$)
m = number of constraints to be penalized
M = number of processing stages in series
M_p = population size
n = number of binary variables
n_i = the number of the batches for product i (for $i = 1, ..., N$)
N = number of products
N_g = number of inequality constraints
N_h = number of equality constraints
N_j = number of parallel units for stage j ($j = 1, ..., M$)
N_{p_i} = the number of parallel lines at level I of the pump network
N_{s_i} = the number of pumps in series at level i of the pump network
P_i = the power requirement at level i of the pump network
Δp_i = the pressure rise at level i at level i of the pump network
P_l = the amount of manipulated variable l (for $l = 1, ..., 6$)
P_m = mutation probability
$P_{r,i}$ = the probability that the variable V_i takes the values in subspace $s_{r,i}$ (for $i = 1, ..., I$ and $r = 1, ..., R$)
q_l = the logarithm of P_l (for $l = 1, ..., 6$)
Q_i = the total production for product i (for $i = 1, ..., N$)
R = number of sections of the variable divided
s_i = uniform random number of variable i ($s_i \in \{-1,+1\}$, $i = 1, ..., I$)
$s_{r,i}$ = subspace of variables (for $i = 1, ..., I$ and $r = 1, ..., R$), $= [L_i^r, U_i^r]$
S_{ij} = constants
S_j = the size of product for stage j (for $j = 1, ..., M$)
t_{ij} = constants
T_{ij} = the time required to process one batch of product i in stage j
T_{Li} = the maximum cycling time for product i (for $i = 1, ..., N$)
U_i' = lower bound of variables V_i (for $i = 1, ..., I$)
V = a vector of variables
v_i = value of variable i in section k (for $i = 1, ..., I$)
$\dot{v}_i$ = the flow rate on each line at level i of the pump network
$V_{i,j}$ = the ith variable of the jth individual
$V_{i,k}$ = a variable of a reference individual
V_i^{Pa} = parents of variable i (for $i = 1, ..., I$)
V_i^{Mut} = variable i after mutation stage (for $i = 1, ..., I$)
$vo_j(V)$ = the amount of violation of jth constraint by the candidate solution V (for $j = 1, ..., m$)
w_l = binary variables (for $l = 1, ..., 6$)
x_i = the fraction of total flow going to level i of the pump network
X = a vector of continuous variables
X_j = the concentration of the metabolite j (for $j = 1, ..., 4$)
Y = a vector of integer variables
$y_j = \ln X_j$
$x_i^U\ y_i^U$ = upper bound on variables x and y
$x_i^L\ y_i^L$ = lower bound on variables x and y
Z = a vector of binary variables
z_i = the existence of level i of the pump network
z_{ij} = binary variables

Greek Letters

α_j, β_j = appropriate cost parameters
ω_i = the rotation speed of all pumps at level i of the pump network
Ω = set of corresponding variables that is selected to be mutated

Literature Cited

(1) Stichlmair, J.; Frey, Th. Mixed-Integer Programming Optimization of Reactive Distillation Processes. *Ind. Eng. Chem. Res.* **2001**, *40*, 5978−5982.

(2) Hatzimanikatis, V.; Floudas, C. A.; Bailey, J. E. Optimization of regulatory architectures in metabolic reaction networks. *Biotechnol. Bioeng.* **1996**, *52* (4), 485−500.

(3) Pozivil, J.; Zd'ansky, M. Application of genetic algorithms to chemical flowshop sequencing. *Chem. Eng. Technol.* **2001,** *24* (4), 327−333.

(4) Cardoso, M. F.; Salcedo, R. L.; Barbosa, D. A simulated annealing approach to the solution of MINLP problems. *Comput. Chem. Eng.* **1997,** *21* (12), 1349−1364.

(5) Grossmann, I. E.; Sargent, R. W. H. Optimum design of multipurpose chemical plants. *Ind. Eng. Chem. Process Des. Dev.* **1979,** *18* (2), 343−348.

(6) Kocis, G. R.; Grossmann, I. E. Global Optimization of Nonconvex Mixed-Integer Nonlinear Programming (MINLP) Problems in Process Synthesis. *Ind. Eng. Chem. Res.* **1988**, *27*, 1407−1421.

(7) Adjiman, C. S.; Androulakis, I. P.; Floudas, C. A. Global optimization of mixed-integer nonlinear problems. *AIChE J.* **2000**, *46*, 1769−1797.

(8) Floudas, C. A.; Pardalos, P. M. *Optimization in Computational Chemistry and Molecular Biology: Local and Global Approaches*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 2000.

(9) Bjork, K.-M.; Nordman, R. Solving large-scale retrofit heat exchanger network synthesis problems with mathematical optimization methods. *Chem. Eng. Process.* **2005**, *44 (*8), 869−876.

(10) Holland, J. H. *Adaptation in Natural and Artificial Systems*; University of Michigan Press: Ann Arbor, MI, 1975.

(11) Yokota, T.; Gen, M.; Li, Y. X. Genetic algorithm for non-linear mixed integer programming problems and its application. *Comput. Ind. Eng.* **1996**, *30* (4), 905−917.

(12) Costa, L.; Oliveira, P. Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems. *Comput. Chem. Eng.* **2001**, *25* (2−3), 257−266.

(13) Barbosa, H. J. C.; Lemonge, A. C. C. A new adaptive penalty scheme for genetic algorithms. *Inf. Sci.* **2003**, *156* (3−4), 215−251.

(14) Burke, E. K.; Elliman, D. G.; Weare, R. F. A hybrid genetic algorithm for highly constrained timetabling problems. In *ICGA Proceedings, 1995,* Proceedings of the 6th International Conference on Genetic Algorithms (ICGA'95), Pittsburgh, PA, July 15−19, 1995; Eshelman, L. J., Ed.; Morgan Kaufmann: San Francisco, CA, 1995; pp 605−610.

(15) Shannon, C. E. A mathematical theory of communication. *Bell Syst. Technol. J.* **1948,** *27*, 379−423, 623−657.

(16) Chen, J.; Wong, D. S. H.; Jang, S.- S. Product and process development using artificial neural-network model and information analysis. *AIChE J.* **1998**, *44* (4), 876−887.

(17) Tsujimura, Y.; Gen, M. Entropy-based genetic algorithm for solving TSP. Presented at the 1998 Second International Conference on Knowledge-Based Intelligent Electronic Systems, 1998.

(18) Yeh, C. W.; Jang, S. S. Chemical Process Model Parameter Estimation Using Information Guided Genetic Algorithm. *J. Chem. Eng. Jpn.* **2006**, *39* (2), 197−210.

(19) Pohlheim, H. GEATbx−Introduction evolutionary algorithms: Overview, methods and operators (version 3.7). Available via the Internet at http://www.geatbx.com/, 2005.

(20) Beasley, D.; Bull, D.; Martin, R. A sequential niche technique for multimodal function optimization. *Evolut. Comput.* **1993**, *1* (2), 101−125.

(21) Mahfoud, S. W. A comparison of parallel and sequential niching techniques. In *ICGA Proceedings, 1995*, Proceedings of the 6th International Conference on Genetic Algorithms (ICGA'95), Pittsburgh, PA, July 15−19, 1995; Eshelman, L. J., Ed.; Morgan Kaufmann: San Francisco, CA, 1995; pp 136−143.

(22) Storn, R.; Price, K. Differential Evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Global Opt.* **1997**, *11*, 341−359.

(23) Angira, R.; Babu, B. V. Optimization of process synthesis and design problems: A modified differential evolution approach. *Chem. Eng. Sci.* **2006**, *61*, 4707−4721.

(24) Rényi, A. *A Diary on Information Theory*; Wiley Series in Probability and Mathematical Statistics; Wiley: New York, 1987.

(25) Mühlenbein, H.; Schlierkamp-Voosen, D. Predictive models for the breeder genetic algorithm: I. continuous parameter optimization. *Evolut. Comput.* **1993**, *1* (1), 25−49.

(26) Lemonge, A. C. C.; Barbosa, H. J. C. An adaptive penalty scheme for genetic algorithms in structural optimization. *Int. J. Numer. Methods Eng.* **2004**, *59*, 703−736.

(27) Ombuki, B. M.; Ventresca, M. Local search genetic algorithms for job shop scheduling problem. *Appl. Intell.* **2004**, *21*, 99−109.

(28) Goyal, V.; Ierapetritou, M. G. Computational studies using a novel simplicial-approximation based algorithm for MINLP optimization. *Comput. Chem. Eng.* **2004**, *28*, 1771−1780.

(29) Westerlund, T.; Pettersson, F.; Grossman, I. E. Optimization of pump configurations as a MINLP Problem. *Comput. Chem. Eng.* **1994**, *18* (9), 845−858.

(30) Floudas, C. A.; Pardalos, P. M.; Adjiman, C. A. et al. *Handbook of Test Problems in Local and Global Optimization*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1999.

Received for review June 7, 2006
Revised manuscript received December 27, 2006
Accepted January 8, 2007

IE060727H