Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/090647

A Heuristic Method for Integer Programming using Discrete Hybrid Evolutionary Algorithm

Hong Li[1,*], *Li Zhang*[1] *and Yong-Chang Jiao*[2]

[1] School of Mathematics and Statistics, Xidian University, Xi'an 710071, P. R. China
[2] Institute of Antennas and EM Scattering, Xidian University, Xi'an 710071, P. R. China

Received: 15 Sep. 2013, Revised: 26 Mar. 2014, Accepted: 28 May 2014
Published online: 1 Nov. 2015

Abstract: A discrete hybrid evolutionary algorithm is developed to solve global numerical optimization problems with discrete variables. In this algorithm, the orthogonal experimental design acts as the crossover operator to achieve crossover, and the migration operator is employed to keep the population's diversity. In addition, the simplified quadratic interpolation method is taken as a local search operator, which is adopted to improve the algorithm's local search ability. Moreover, a few of foreign chromosomes, which are generated via randomly perturbing the best candidate chromosome in the current population, are introduced into the next generation to avoid most of chromosomes gradually clustering around the best candidate chromosome in some subsequent generations. A rounding and truncation procedures is incorporated in the operations of the algorithm to ensure that the integer restrictions and box constraints are satisfied. Numerical experiments on 22 test problems have demonstrated the efficiency of the proposed method.

Keywords: integer programming, discrete evolutionary algorithm, orthogonal experimental design, discrete global optimization

1 Introduction

Evolutionary algorithm (EA), based on the principles of natural biological evolution, is a class of stochastic optimization search method. During the last decades, EA has received a considerable attraction and has experienced a rapid development. EA has been shown to be a global and robust method for solving highly nonlinear, nondifferentiable, and multimodal optimization problems, which means that traditional, gradient-based optimization algorithms fail and stochastic optimization techniques must be employed [1].

Generally speaking, all population-based optimization algorithms suffer from long computational times because of their evolutionary or stochastic nature [2]. Several studies have shown that incorporating some form of domain knowledge can greatly improve the search capability of EAs [2][3]. Many problem dependent heuristics, such as approximation algorithm, local search techniques, specialized recombination operators, etc., have been tried in many different ways to accomplish this task. In particular, the hybridization of EAs with local searches has proven to be very promising [3].

Most of the ongoing research in the global optimization centers on the development of algorithms for solving the continuous problem, in which the decision variables are restricted to real-value (continuous) variables (see, e.g., [3][4]). However, it is much difficult to solve integer programming problems that involve discrete variables. This is mainly because these problems pose major algorithmic challenges in the development of effective solution strategies. Especially, it is an algorithm challenge to locate the global optima of large-dimensional integer programming problems. Generally speaking, the efficient methods designed for continues optimization problems can not be directly used to solve integer programming problems. However, an integer programming problem with bounded discrete variables, especially binary variables, may be transformed into an equivalent continuous global optimization problem, such that the problem can be solved by the methods of continuous global optimization [5][6]. Unfortunately, the transformed problems usually have large numbers of local minimizers, making them harder to solve than typical global optimization problems [6].

Stochastic or heuristic approaches can be applied to almost all discrete optimization problems [7], especially

[*] Corresponding author e-mail: lihong@mail.xidian.edu.cn

© 2015 NSP
Natural Sciences Publishing Cor.

for high-dimensional cases [8]. This is mainly because stochastic approaches need not depend on some analytical properties of problems. There exist some stochastic approaches to solve discrete optimization problems, such as the simulated annealing algorithms [9][10], the evolutionary algorithms [11][12][13][14], the tabu search algorithms [15], particle swarm optimization [16][17], and controlled random search technique [18].

In this paper, we develop a discrete hybrid evolutionary algorithm (DHEA) to deal with the integer programming problems. The corresponding evolutionary operations, i.e. mutation, crossover, and selection, are discussed in detail. For the performance enhancement, a migration operation and a local search strategy are introduced. The performance of the proposed DHEA is assessed by carrying out optimization on 22 test problems. Compared to some available algorithms such as PSO [16], RST2ANU [18], and DDCM [8], DHEA is shown to be able to achieve more accurate results and yet with a much reduced computational effort in almost all the cases examined. It is shown that DHEA maintains a superior effectiveness for such problems.

This paper is organized as follows. The integer programming is formulated in the next section. A discrete hybrid evolutionary algorithm is proposed for solving integer programming problems in Section 3. Computational results of DHEA and comparisons with other available algorithms are provided in Section 4. Finally, we conclude this paper and give the future work prospect in Section 5.

2 Problem formulation

This paper considers the general integer programming problem formulated as

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && g_j(x) \le 0,\ j=1,2,\cdots,p \\ & && x \in X \subset \mathscr{Z}^n \end{aligned} \tag{1}$$

where $\mathscr{Z}^n$ is the set of integer points in $\mathscr{R}^n$, the set X is box constraints, i.e.

$$X = \{x \in \mathscr{Z}^n | x_i^L \le x_i \le x_i^U,\ x_i^L \text{ and } x_i^U \in \mathscr{Z}\}.$$

In most practical applications the integer bounds x^L and x^U are available or can be easily specified.

If problem (1) contains equality constraint $h(x) = 0$, it may be approximated by inequality constraint $|h(x)| - \delta \le 0$, where δ stands for the degree of violation.

The objective function $f(x)$ and constraint functions $g_j(x)$, $j = 1,2,\cdots,p$, may be linear or nonlinear. The feasible integer points set $\mathscr{F}$ is defined by

$$\mathscr{F} = \{x \in X \subset \mathscr{Z}^n | g_j(x) \le 0,\ j = 1,2,\cdots,p\}. \tag{2}$$

An integer point $x^* \in \mathscr{F}$ is called a discrete global optimal solution of problem (1), if $f(x) \ge f(x^*)$, for all $x \in \mathscr{F}$.

3 A discrete hybrid evolutionary algorithm for problem (1)

In this section, a novel discrete hybrid evolutionary algorithm (DHEA) is introduced to deal with integer programming problems. We present the following several enhancements in the DHEA for global optimization with discrete variables.

(1) An integer coding technique is applied to optimization problems with discrete variables. In the integer coding representation, each chromosome is encoded as a vector of integer numbers, with the same length as the vector of decision variables.

(2) The dynamic direction mutation operator is adopted. This mutation operator can explore the region of the optimum offspring.

(3) The two tools of the orthogonal experimental design, two-level orthogonal array and factor analysis, are employed in the crossover operator. The systematic reasoning ability of the orthogonal experimental design is used to economically identify the potentially better one of two genes of each pair, and then obtain a potentially good approximation to the best one of all combinations by executing fewer experiments.

(4) Through the crossover, the population diversity and its exploration of the search space are rapidly decreased, and the clustered chromosomes cannot reproduce newly better offspring. In order to greatly increase the exploration of the search space and decrease the selection pressure for a small population, a migration operator is introduced to regenerate a newly diverse population of chromosomes.

(5) The simplified three-point quadratic interpolation method is taken as a local search operator, which is adopted to improve the algorithm's local search ability, and to prevent the algorithm from getting trapped in the local optima.

(6) A few of foreign chromosomes, which are generated via randomly perturbing the best candidate chromosome in the current population, are introduced into the next generation to avoid most of chromosomes gradually clustering around the best candidate chromosome in some subsequent generations.

(7) A rounding and truncation procedures is incorporated in the operations of the algorithm to ensure that the integer restrictions imposed on the decision variables and box constraints are satisfied.

By using these strategies, DHEA has high performance in solving the benchmark problems comprising many variables, as compared with some existing algorithms. The details of the components in DHEA are provided as follows.

3.1 Chromosome coding and Initialization

For solving integer programming problem (1) by the discrete hybrid evolutionary algorithm, a chromosome is represented by an integer vector, i.e. a vector $x = (x_1, x_2, \cdots, x_n)^\top$ is used to express a chromosome, where $x_i \in \mathscr{Z}$, $i = 1, 2, \cdots, n$.

We randomly generate an n-dimensional integer vector as an initial chromosome. By the following `Algorithm 1`, generate M chromosomes such that they should try to cover the entire search space uniformly, where M denotes the population size.

`Algorithm 1`:

Step 1. Generate an $n \times n$ random diagonal matrix $\gamma = \text{diag}(\gamma_1, \cdots, \gamma_i, \cdots, \gamma_n)$, where $\gamma_i \in (0,1)$, $i = 1, 2, \cdots, n$.

Step 2. Let $x = \text{INT}[x^L + \gamma \cdot (x^U - x^L)]$, where the operator $\text{INT}[t]$ is expressed as the nearest integer vector to the real vector t.

Step 3. Repeat the above steps M times and produce M initial chromosomes such that they constitute the initial population.

3.2 Fitness function

The penalty function method, due to its simplicity, is by far the most widely studied and used in handling constraints [23]. Based on the exact penalty method, the fitness function $fit(x)$ is expressed as:

$$fit(x) = \begin{cases} f(x), & \text{if } x \in \mathscr{F} \\ f(x) + \Theta \cdot P(x), & \text{otherwise} \end{cases} \tag{3}$$

where Θ $(\Theta > 0)$ is the penalty parameter, and $P(x) = \sum_{j=1}^{p} \max\{g_j(x), 0\}$.

3.3 Mutation operation

Differential evolution (DE) has been used in many practical cases and has demonstrated good convergence properties. The essential ingredient in the DE is mutation operator. The difference of two random vectors acts as a search direction in the solution space. The mutation factor selected between zero and one is used to control the search step. Due to the integer restriction imposed on the decision variable, the real perturbation for the variable is then rounded to the nearest integer number. As a result, the each component of mutant chromosome becomes the integer number. In this paper we adopt the mutation operator from DE similar to that in [1] (see Eqn.(8) in [1]).

Let p_m be the mutation probability. Denote the best chromosome, a chromosome with minimum fitness value among all chromosomes generated till now, by $x^b = (x_1^b, x_2^b, \cdots, x_n^b)^\top$. In current population, randomly select three different chromosomes with probability p_m, denoted by $x^r = (x_1^r, x_2^r, \cdots, x_n^r)^\top$, $x^p = (x_1^p, x_2^p, \cdots, x_n^p)^\top$ and $x^q = (x_1^q, x_2^q, \cdots, x_n^q)^\top$.

Generate a child mutation chromosome $x^m = (x_1^m, x_2^m, \cdots, x_n^m)^\top$ by

$$x^m = \min\Big\{\max\Big\{\text{INT}[x^r + \alpha \cdot (x^b - x^r) + \beta \cdot (x^p - x^q)], x^L\Big\}, x^U\Big\}, \tag{4}$$

where $\alpha = \text{diag}(\alpha_1, \alpha_2, \cdots, \alpha_n)$, $\beta = \text{diag}(\beta_1, \beta_2, \cdots, \beta_n)$, α_i and β_i $(i = 1, 2, \cdots, n)$ are real random numbers between zero and one. The operator $\text{INT}[t]$ is expressed as the nearest integer vector to the real vector t.

In Equation (4), a real vector is generated via using the direction mutation operator, and then this real vector is rounded to the nearest integer vector. For some components of this integer vector beyond the bounds, by truncation procedure, keep them within the bounds. For example, if the ith component of this integer vector is less than the lower bound x_i^L, let the ith component be x_i^L; if the ith component is larger than the upper bound x_i^U, let the ith component be x_i^U.

3.4 Crossover operation

Crossover offspring are generated by using orthogonal experimental design with both orthogonal array and factor analysis. An efficient way to study the effect of several factors simultaneously is to use orthogonal experimental design. The factors are the variables, which affect response variables, and a setting of a factor is regarded as a level of the factor. A “complete factorial” experiment would make measurements at each of all possible level combinations. However, the number of level combinations is often so large that this is impractical, and a subset of level combinations must be judiciously selected to be used, resulting in a “fractional factorial” experiment. Orthogonal experimental design utilizes properties of fractional factorial experiments to efficiently determine the best combination of factor levels to use in design problems [24][25] .

A two-level orthogonal array similar to that in [24] is used in this paper. The general symbol for two-level standard orthogonal arrays is $\text{L}_N(2^{N-1})$, where N is the number of experimental runs, $N-1$ denotes the number of factors. Let $n \le N-1$. If $n = N-1$, we directly adopt the standard orthogonal array $\text{L}_N(2^{N-1})$. If $n < N-1$, we adopt only the first n columns of the standard orthogonal array $\text{L}_N(2^{N-1})$ or the available orthogonal array $\text{L}_N(2^n)$ from source. According to the problem dimension, we choose appropriate two-level orthogonal array $\text{L}_N(2^{N-1})$ to execute the matrix experiments, and generate N offspring chromosomes. The algorithm of constructing orthogonal arrays can be found in [20]. Many existing orthogonal arrays can also be obtained in [19]. Several orthogonal arrays are available from a library of orthogonal arrays at http://neilsloane.com/oadir/.

After evaluation of each chromosome in the N combinations, the summarized data are analyzed using the factor analysis. Factor analysis can evaluate the effects of individual factors on the fitness value, rank the most effective factors, and determine the better level for each factor. In this paper, the factor analysis in [4] is adopted in our algorithm. Let fit_i denote a fitness value of the combination corresponding to the experiment i, $i = 1, 2, \cdots, N$, where N is the total number of experiments. Define the main effect of factor j with level k as E_{jk} where $j = 1, 2, \cdots, n$ and $k = 1, 2$:

$$E_{jk} = \sum_{i=1}^{N} fit_i \cdot \chi_i \tag{5}$$

where $\chi_i = 1$ if the level of factor j of experiment i is k; otherwise, $\chi_i = 0$. Considering the case that the problem is to be minimized (smaller-the-better), the level 1 of factor j makes a better contribution to the fitness value than level 2 of factor j does when $E_{j1} < E_{j2}$, so the better level is level 1 for factor j. If $E_{j1} > E_{j2}$, level 2 is the better one. If $E_{j1} = E_{j2}$, levels 1 and 2 have the same contribution.

After the better level for each factor is selected, a best chromosome can also be obtained. This best chromosome has the lowest fitness value among those of parents and N combinations of factor levels, where N is the total number of experiments. The purpose for the use of the orthogonal experimental design is to produce a potentially best chromosome from two randomly selected parents chromosomes.

3.5 Migration operation

In order to improve the population's diversity and avoid the premature convergence, we adopt a migration operator similar to that in [11].

Let p_{mi} be the migration probability. A chromosome is randomly selected from the candidate chromosomes generated by the crossover operation with probability p_{mi}, denoted by $x^c = (x_1^c, x_2^c, \cdots, x_n^c)^\top$. A new chromosome $x^v = (x_1^v, x_2^v, \cdots, x_n^v)^\top$ is generated according to:

$$x_k^v = \begin{cases} x_k^c + \text{INT}[\rho_k \cdot (x_k^L - x_k^c)], & \text{if } \tau_k < \frac{x_k^c - x_k^L}{x_k^U - x_k^L} \\ x_k^c + \text{INT}[\rho_k \cdot (x_k^U - x_k^c)], & \text{otherwise} \end{cases} \tag{6}$$

$$k = 1, 2, \cdots, n.$$

where ρ_k and τ_k $(k = 1, 2, \cdots, n)$ are random numbers between zero and one.

3.6 Local search operation

In this subsection, we introduce the simplified three-point quadratic interpolation method [18], which is adopted to improve the local search ability of the main algorithm, and to prevent the main algorithm from getting trapped in the local optima.

Denote the best chromosome corresponding to the minimum fitness value by $x^1 = (x_1^1, \cdots, x_n^1)^\top$, and choose randomly two other distinct chromosomes $x^2 = (x_1^2, \cdots, x_n^2)^\top$ and $x^3 = (x_1^3, \cdots, x_n^3)^\top$. Let $f_1 = fit(x^1)$, $f_2 = fit(x^2)$, $f_3 = fit(x^3)$, then determine the approximate minimal chromosome $\bar{x} = (\bar{x}_1, \cdots, \bar{x}_n)^\top$ using the three-point quadratic interpolation formula:

$$\bar{x}_i = \frac{A_i}{2B_i}, i = 1, 2, \cdots, n. \tag{7}$$

where $A_i = [(x_i^1)^2 - (x_i^2)^2] f_3 + [(x_i^2)^2 - (x_i^3)^2] f_1 + [(x_i^3)^2 - (x_i^1)^2] f_2$, $B_i = (x_i^1 - x_i^2) f_3 + (x_i^2 - x_i^3) f_1 + (x_i^3 - x_i^1) f_2$. Note that $\bar{x}$ is the extremal point of the quadratic curve passing through x^1, x^2, x^3, and $\bar{x}$ is a vector of real variables. In order to make $\bar{x}$ satisfy the integer restrictions and box constraints, we modify $\bar{x}$ using the rounding and truncation procedures, and generate a trial chromosome x^a:

$$x^a = \min\left\{\max\left\{\text{INT}[\bar{x}], x^L\right\}, x^U\right\}. \tag{8}$$

3.7 Selection operation

We first compare the fitness values of all the chromosomes, including those in the current population and all the offspring, rearrange all the chromosomes in the ascending order of their fitness values, and find the best chromosome $x^b = (x_1^b, x_2^b, \cdots, x_n^b)^\top$ with the minimum fitness value. We choose the first $M - M_1$ chromosomes, and then randomly generate M_1 new chromosomes $x^d = (x_1^d, x_2^d, \cdots, x_n^d)^\top$ by:

$$x_j^u = x_j^b + \text{INT}[-1 + \text{rand}(0,1) * 2], \tag{9}$$

$$x_j^d = \begin{cases} x_j^L, & \text{if } x_j^u \le x_j^L \\ x_j^u, & \text{if } x_j^L < x_j^u < x_j^U \\ x_j^U, & \text{if } x_j^u \ge x_j^U \end{cases} \tag{10}$$

$$j = 1, 2, \cdots, n.$$

where rand$(0,1)$ represents a random number between zero and one. The resulting M chromosomes constitute the next population. In addition, we retain the best chromosome of every generation.

3.8 Stopping criterion

If the optimal solution is reached first, then the algorithm stops; otherwise, the algorithm is executed to the maximal number of generations *MaxG*, then the algorithm stops and the best chromosome with minimum fitness value will be then taken as a global minimizer of problem.

Now the discrete hybrid evolutionary algorithm for problem (1) is presented as follows.

`Algorithm 2`:

Step 0. Parameter Setting
Choose population size M, mutation probability p_m, cross-over probability p_c, migration probability p_{mi}, the degree of violation δ, penalty parameter Θ, number of the foreign chromosomes introduced into the next population M_1, suitable two-level orthogonal array $\mathrm{L}_N(2^{N-1})$, and the maximal number of generations $MaxG$.

Step 1. Initialization
Let $t = 0$, an initial population $P(t)$ is generated by the `Algorithm 1`, and evaluate the fitness values of initial chromosomes in the $P(t)$.

Step 2. Mutation operation
Step 2.1. Determine the best chromosome with minimum fitness value among all chromosomes generated till now.
Step 2.2. Choose randomly three different chromosomes in the current population with probability p_m.
Step 2.3. Generate a mutant chromosome by using Equation (4), and evaluate its fitness value.
Step 2.4. Repeat Steps 2.2-2.3 M times, produce $M_m(= p_m * M)$ mutant chromosomes, which form a temporary population $P_m(t)$.

Step 3. Crossover operation
Step 3.1. Choose randomly two chromosomes from the population $P_m(t)$ with probability p_c to execute the matrix experiment.
Step 3.2. Evaluate the fitness values of the chromosomes generated by the matrix experiment.
Step 3.3. Calculate the effects of the various factors (E_{j1}, E_{j2}), $j = 1,2,\cdots,n$, thus generate a best chromosome, and then evaluate its fitness value.
Step 3.4. Repeat Steps 3.1-3.3 M_m times, produce $M_c(= p_c * M_m)$ offspring chromosomes, which form a temporary population $P_c(t)$.

Step 4. Migration operation
Step 4.1. Choose randomly a chromosome from the temporary population $P_c(t)$ with probability p_{mi}.
Step 4.2. Generate a new chromosome by using Equation (6), and evaluate its fitness value.
Step 4.3. Repeat Steps 4.1-4.2 M_c times, produce $M_{mi}(= p_{mi} * M_c)$ offspring chromosomes, which form a temporary population $P_{mi}(t)$.

Step 5. Local search operation
Step 5.1. Rearrange all the chromosomes in the current population and all the temporary populations $(P(t) \cup P_m(t) \cup P_c(t) \cup P_{mi}(t))$ in ascending order of their fitness values, and choose the first M chromosomes to form a trial population $P'(t)$.
Step 5.2. Find the best chromosome x^1 with minimal fitness value $fit(x^1)$, and the worst chromosome x^w with maximal fitness value $fit(x^w)$ in the population $P'(t)$. Randomly choose two other distinct chromosomes from the population $P'(t)$, denoted by x^2, and x^3, respectively. Let $f_1 = fit(x^1)$, $f_2 = fit(x^2)$, and $f_3 = fit(x^3)$.
Step 5.3. For some $i \in \{1,2,\cdots,n\}$, if $(x_i^1 - x_i^2)f_3 + (x_i^2 - x_i^3)f_1 + (x_i^3 - x_i^1)f_2 < \varepsilon$ $(\varepsilon = 10^{-4})$, then let $x^a = x^w$, and $fit(x^a) = fit(x^w)$, go to Step 5.5; Otherwise, go to Step 5.4.
Step 5.4. Calculate x^a by using the Equations (7)(8), and then the fitness value $fit(x^a)$.
Step 5.5. If $fit(x^a) < fit(x^w)$, then replace the worst chromosome x^w in the population $P'(t)$ with x^a and form a new population $P''(t)$; Otherwise, keep the population $P'(t)$ unchangeable and the new population $P''(t) = P'(t)$.

Step 6. Selection operation
Step 6.1. Choose the first $M - M_1$ chromosomes from the population $P''(t)$ based on the ascending order of fitness values.
Step 6.2. Generate randomly M_1 chromosomes according to Equations (9)(10).
Step 6.3. The resulting M chromosomes constitute the next population $P(t+1)$. In addition, we retain the best chromosome of every generation.

Step 7. Stopping criterion
If the stopping criterion is met, then stop, and output the best chromosome with minimal fitness value. This chromosome is taken as the global optimal solution of problem (1). Otherwise, set $t = t+1$, and go to Step 2.

4 Computational experiences and comparison

To demonstrate feasibility and efficiency of the proposed algorithm (DHEA), we test 22 integer programming. The test problems are selected from the literature [8] [16] [18]. The details of these problems are provided in the Appendix.

The proposed algorithm is programmed in Matlab 7.0 for working on a microcomputer with Intel Pentium IV/3.06 GHz CPU and 1GB RAM.

During the simulations, we adopted the following parameter settings of the DHEA for each test problem.
Population size: $M = 50$ for $n < 50$; $M = n$ for $n \geq 50$;
Mutation probability: $p_m = 0.3$;
Crossover probability: $p_c = 0.8$;
Migration probability: $p_{mi} = 0.4$;
Degree of violation: $\delta = 0.0001$;
Penalty parameter: $\Theta = 10000$;
Number of the foreign chromosomes introduced into the next population: $M_1 = 5$;
The maximal number of generations $MaxG = 1500$ for each test problem.

Two-level orthogonal arrays used in this paper are provided as follows:
$\mathrm{L}_4(2^3)$ for 2-dimensional or 3-dimensional problem;
$\mathrm{L}_8(2^7)$ for 4-dimensional or 5-dimensional problem;
$\mathrm{L}_{12}(2^{11})$ for 10-dimensional problem;
$\mathrm{L}_{16}(2^{15})$ for 13-dimensional problem;
$\mathrm{L}_{32}(2^{31})$ for 25-dimensional or 30-dimensional problem;
$\mathrm{L}_{64}(2^{63})$ for 40-dimensional or 50-dimensional problem;

$L_{200}(2^{100})$ for the problem with 100 dimensions;
$L_{400}(2^{200})$ for the problem with 200 dimensions.

The performance of the DHEA is evaluated based on the optimization results on the 22 test problems as compared with some existing discrete global optimization algorithms. For each test problem, 20 independent runs with different seeds from the random number generator are performed to observe the consistency of the outcome.

The computational results of DHEA are summarized in Table 1, including the following:

(1) Success rate (SR), i.e., the proportion of convergence to the global optimum;

(2) Mean number of fitness evaluations (FES) for the algorithm to stop at Step 7.

From Table 1, it is obvious that DHEA can find the global optimal solutions with high success rate for all problems. It indicates that DHEA is robust and stable in finding an optimal solution. The computational cost of DHEA can be measured by computing the mean number of fitness evaluations. As described in Table 1, DHEA requires a small number of fitness evaluations. On the whole, DHEA exhibits a consistent and satisfactory performance in terms of the success rate, as well as the computational cost. It is, therefore, concluded that DHEA is a promising technique in performing discrete global optimization for future practical applications.

We compared our approach against the following three approaches:

(1) Particle Swarm Optimization (PSO) [16]: PSO is an evolutionary computing scheme; it explores the insect swarm behavior. Each particle of the swarm was truncated to the closest integer, after the determination of its new position.

(2) Controlled Random Search Technique Incorporating the Simulated Annealing Concept (RST2ANU algorithm)[18]: This algorithm, which primarily is based on the original controlled random search approach of Price, incorporates a simulated annealing type acceptance criterion in its working so that not only downhill moves but also occasional uphill moves can be accepted. In its working it employs a special truncation procedure which not only ensures that the integer restrictions imposed on the decision variables are satisfied, but also creates greater possibilities for the search leading to a global optimal solution.

(3) Discrete Dynamic Convexized Method (DDCM)[8]: An auxiliary function is constructed, which has the same discrete global minimizers as the problem. The minimization of the function using a discrete local search method can escape from previously converged discrete local minimizers by taking increasing values of a parameter. The algorithm called discrete dynamic convexized method is proposed for original problem by minimizing the auxiliary function.

Since the optimization results from the literature using these algorithms are available only for some of the 22 selected test problems, the comparison will be made accordingly. The comparative results are summarized in Table 1.

From the literature [16] the optimization results using PSO are available only for seven of the 22 test problems which are problems $1-7$. For all experiments the initial population was only taken randomly inside $[-100,100]^n$ by the DHEA, where n is the corresponding dimension. Table 1 presents the optimization results obtained by the DHEA in comparison with those from the PSO. As can be seen, the DHEA is able to find the global optimal solutions for the 7 test problems with relatively high success rate, indicating that the algorithm is both effective and statistically stable. Comparing with the PSO, the DHEA achieves generally much higher success rate except for problem 4, while the required computational effort is reduced considerably. The comparative study shows that the performance of DHEA is superior to that of the PSO.

From the literature [18] the optimization results using RST2ANU are available only for eight of the 22 test problems which are problems $8-15$. From Table 1, it can be seen that DHEA can achieve satisfied results for the listed problems. On average, the computational cost required by DHEA is much less than that required by RST2ANU. For the problem 8, RST2ANU provided the known maximal function value 303062432, and the DHEA can determine the better maximal function value 304148583 with high success rate. For the problem 11, RST2ANU provided the known maximal function value 1030361, and the DHEA can identify the better maximal function value 1352439 with high success rate. For the problem 13, RST2ANU reported a maximum solution with function value 209205232, and the optimal function value obtained using the DHEA is 216300719. For the remaining problems, once again, the DHEA exhibits a superior performance.

From the literature [8] the optimization results using the DDCM are available only for eleven of the 22 test problems, which are problems $12-22$. For high-dimension problem 16, DHEA can optimize the problems with up to 200 dimensions. From Table 1, it can be seen that DHEA can achieve global optimization results for problem 16, while the required computational effort is small. For small-dimension problems, DHEA is also able to obtain global minimum solutions with lower computational cost except for problems $13-15$ and 17.

5. Conclusion and future work

In this paper, a discrete hybrid evolutionary algorithm is developed to globally solve integer programming problems, including nonlinear integer programming problems, linear integer programming problems as well as 0-1 integer programming problems. The behavior of the proposed algorithm seems to be stable even for high dimensional cases, exhibiting fairly high success rates even with modest population sizes. Experiments on 22

Table 1: Comparison of DHEA with other algorithms in the literature for benchmark problems.

		DHEA		PSO		RST2ANU		DDCM	
Problem	n	SR (%)	FES	SR (%)	FES	SR (%)	FES	SR (%)	FES
1	25	100	1276.8	100	171110.0	-	-	-	-
1	30	100	1288.7	80	274692.0	-	-	-	-
2	25	100	1271.3	100	171610.0	-	-	-	-
2	30	100	1283.7	84	274284.0	-	-	-	-
3	5	85	16938.4	80	75060.0	-	-	-	-
4	2	85	1652.4	100	8066.4	-	-	-	-
5	2	100	816.0	100	8321.6	-	-	-	-
6	2	100	975.6	100	8388.0	-	-	-	-
7	4	100	1021.9	92	18476.8	-	-	-	-
8	100	70	410335.5	-	-	100	46617	-	-
9	30	100	610.5	-	-	100	26042	-	-
10	10	95	1069.3	-	-	100	279255	-	-
11	40	100	759.6	-	-	100	150562	-	-
12	5	100	5571.0	-	-	100	187794	100	24679.2
13	10	100	24388.8	-	-	100	2695	100	13655.1
14	13	100	2237.4	-	-	100	7887	100	658.1
15	3	90	5229.2	-	-	70	206	100	689.7
16	25	90	13932.3	-	-	-	-	100	28089.6
16	50	85	34960.1	-	-	-	-	100	180368.3
16	100	85	203659.0	-	-	-	-	100	1559704
16	200	100	1204337.7	-	-	-	-	100	10234584
17	4	100	1020.1	-	-	-	-	100	752.8
18	2	75	5320.9	-	-	-	-	100	756941.2
19	2	100	958.1	-	-	-	-	100	12780839
20	4	100	2101.2	-	-	-	-	100	16622286
21	2	100	969.3	-	-	-	-	100	55685.9
22	3	100	3157.5	-	-	-	-	100	7055.2

test problems show that the proposed algorithm is able to find the global optimal solution in all cases. Compared with other algorithms, the proposed algorithm achieves a considerable reduction of the computational effort. Experimental results indicate that the proposed algorithm is an efficient and robust method and should be considered as a good alternative to handle integer programming problems.

One issue that deserves some further study is the sensitivity of our approach to parameters such as population size, mutation probability, crossover probability, migration probability, and number of the foreign chromosomes introduced into the population. Another path of future research consists of modifying relevantly this discrete hybrid evolutionary algorithm to solve mixed integer programming problems. Additionally, we are also interested in applying our approach to the solution of real-world optimization problems.

Acknowledgments

This work was supported by Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2013JM1022), and the Fundamental Research Funds for the Central Universities (Grant No. K50511700004).

The authors are grateful to the anonymous referee for helpful comments that improved this paper.

Appendix

Problem 1[16]:

$$\begin{aligned} &\min\ f(x) = |x_1| + |x_2| + \cdots + |x_n| \\ &\text{s.t.}\ -100 \le x_i \le 100,\ x_i \in \mathscr{Z},\ i = 1,2,\cdots,n \end{aligned}$$

The global minimum solution is $x_i^* = 0, i = 1,2,\cdots,n$, with $f(x^*) = 0$.

Problem 2[16]:

$$\begin{aligned} &\min\ f(x) = x_1^2 + x_2^2 + \cdots + x_n^2 \\ &\text{s.t.}\ -100 \le x_i \le 100,\ x_i \in \mathscr{Z},\ i = 1,2,\cdots,n \end{aligned}$$

The global minimum solution is $x_i^* = 0, i = 1,2,\cdots,n$, with $f(x^*) = 0$.

Problem 3[16]:

$$\begin{aligned} &\min\ f(x) = -Ax + x^\top Bx \\ &\text{s.t.}\ -100 \le x_i \le 100,\ x_i \in \mathscr{Z},\ i = 1,2,3,4,5 \end{aligned}$$

where $A = \begin{bmatrix} 15 \\ 27 \\ 36 \\ 18 \\ 12 \end{bmatrix}^\top$, $B = \begin{bmatrix} 35 & -20 & -10 & 32 & -10 \\ -20 & 40 & -6 & -31 & 32 \\ -10 & -6 & 11 & -6 & -10 \\ 32 & -31 & -6 & 38 & -20 \\ -10 & 32 & -10 & -20 & 31 \end{bmatrix}$.
The global minimum solutions are $x^* = (0,11,22,16,6)^\top$ and $x^* = (0,12,23,17,6)^\top$, with $f(x^*) = -737$.

Problem 4[16]:

$$\begin{aligned} \min\ & f(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 \\ \text{s.t.}\ & -100 \le x_i \le 100,\ x_i \in \mathscr{Z},\ i = 1,2 \end{aligned}$$

The global minimum solution is $x^* = (3,2)^\top$ with $f(x^*) = 0$.

Problem 5[16]:

$$\begin{aligned} \min\ & f(x) = (9x_1^2 + 2x_2^2 - 11)^2 + (3x_1 + 4x_2^2 - 7)^2 \\ \text{s.t.}\ & -100 \le x_i \le 100,\ x_i \in \mathscr{Z},\ i = 1,2 \end{aligned}$$

The global minimum solution is $x^* = (1,1)^\top$ with $f(x^*) = 0$.

Problem 6[16]:

$$\begin{aligned} \min\ & f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \\ \text{s.t.}\ & -100 \le x_i \le 100,\ x_i \in \mathscr{Z},\ i = 1,2 \end{aligned}$$

The global minimum solution is $x^* = (1,1)^\top$ with $f(x^*) = 0$.

Problem 7[16]:

$$\begin{aligned} \min\ & f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 \\ & \quad + 10(x_1 - x_4)^4 \\ \text{s.t.}\ & -100 \le x_i \le 100,\ x_i \in \mathscr{Z},\ i = 1,2,3,4 \end{aligned}$$

The global minimum solution is $x^* = (0,0,0,0)^\top$ with $f(x^*) = 0$.

Problem 8[18]:

$$\begin{aligned} \max\ & f(x) = 50x_1 + 150x_2 + 100x_3 + 92x_4 + 55x_5 \\ & + 12x_6 + 11x_7 + 10x_8 + 8x_9 + 3x_{10} + 114x_{11} \\ & + 90x_{12} + 87x_{13} + 91x_{14} + 58x_{15} + 16x_{16} \\ & + 19x_{17} + 22x_{18} + 21x_{19} + 32x_{20} + 53x_{21} \\ & + 56x_{22} + 118x_{23} + 192x_{24} + 52x_{25} + 204x_{26} \\ & + 250x_{27} + 295x_{28} + 82x_{29} + 30x_{30} + 29x_{31}^2 \\ & - 2x_{32}^2 + 9x_{33}^2 + 94x_{34} + 15x_{35}^3 + 17x_{36}^2 - 15x_{37} \\ & - 2x_{38} + x_{39} + 3x_{40}^4 + 52x_{41} + 57x_{42}^2 - x_{43}^2 \\ & + 12x_{44} + 21x_{45} + 6x_{46} + 7x_{47} - x_{48} + x_{49} \\ & + x_{50} + 119x_{51} + 82x_{52} + 75x_{53} + 18x_{54} + 16x_{55} \\ & + 12x_{56} + 6x_{57} + 7x_{58} + 3x_{59} + 6x_{60} + 12x_{61} \\ & + 13x_{62} + 18x_{63} + 7x_{64} + 3x_{65} + 19x_{66} + 22x_{67} \\ & + 3x_{68} + 12x_{69} + 9x_{70} + 18x_{71} + 19x_{72} + 12x_{73} \\ & + 8x_{74} + 5x_{75} + 2x_{76} + 16x_{77} + 17x_{78} + 11x_{79} \\ & + 12x_{80} + 9x_{81} + 12x_{82} + 11x_{83} + 14x_{84} + 16x_{85} \\ & + 3x_{86} + 9x_{87} + 10x_{88} + 3x_{89} + x_{90} + 12x_{91} \\ & + 3x_{92} + 12x_{93} - 2x_{94}^2 - x_{95} + 6x_{96} + 7x_{97} \\ & + 4x_{98} + x_{99} + 2x_{100} \\ \text{s.t.}\ & \textstyle\sum_{i=1}^{100} x_i \le 7500,\ \sum_{i=1}^{50} 10x_i + \sum_{i=1}^{100} x_i \le 42000 \\ & 0 \le x_i \le 99,\ x_i \in \mathscr{Z},\ i = 1,2,\cdots,100 \end{aligned}$$

The best known function value given in [18] is $f(x^*) = 303062432$.

Problem 9[18]:

$$\begin{aligned} \min\ & f(x) = 1 - \exp[(-1/60)\textstyle\sum_{i=1}^{30} x_i^2] \\ \text{s.t.}\ & 0 \le x_i \le 5,\ x_i \in \mathscr{Z},\ i = 1,2,\cdots,30 \end{aligned}$$

The global minimum solution is $x^* = (0,0,\cdots,0)^\top$ with $f(x^*) = 0$.

Problem 10[18]:

$$\begin{aligned} \min\ & f(x) = C^\top x - 0.5x^\top Qx \\ \text{s.t.}\ & Ax \le B,\ x_i \in \{0,1\},\ i = 1,2,\cdots,10 \end{aligned}$$

where $A = \begin{bmatrix} -2 & -6 & -1 & 0 & -3 & -3 & -2 & -6 & -2 & -2 \\ 6 & -5 & 8 & -3 & 0 & 1 & 3 & 8 & 9 & -3 \\ -5 & 6 & 5 & 3 & 8 & -8 & 9 & 2 & 0 & -9 \\ 9 & 5 & 0 & -9 & 1 & -8 & 3 & -9 & -9 & -3 \\ -8 & 7 & -4 & -5 & -9 & 1 & -7 & -1 & 3 & -2 \end{bmatrix}$,
$B^\top = (-4,22,-6,-23,-12)$, $Q = 100 \times I_{10}$,
$C^\top = (48,42,48,45,44,41,47,42,45,46)$.
The global minimum solution is $x^* = (1,0,0,1,1,1,0,1,1,1)^\top$ with $f(x^*) = -39$.

Problem 11[18]:

$$\begin{aligned} \max\ & f(x) = 215x_1 + 116x_2 + 670x_3 + 924x_4 + 510x_5 \\ & + 600x_6 + 424x_7 + 942x_8 + 43x_9 + 369x_{10} \\ & + 408x_{11} + 52x_{12} + 319x_{13} + 214x_{14} + 851x_{15} \\ & + 394x_{16} + 88x_{17} + 124x_{18} + 17x_{19} + 779x_{20} \\ & + 278x_{21} + 258x_{22} + 271x_{23} + 281x_{24} + 326x_{25} \\ & + 819x_{26} + 485x_{27} + 454x_{28} + 297x_{29} + 53x_{30} \\ & + 136x_{31} + 796x_{32} + 114x_{33} + 43x_{34} + 80x_{35} \\ & + 268x_{36} + 179x_{37} + 78x_{38} + 105x_{39} + 281x_{40} \\ \text{s.t.}\ & 9x_1 + 11x_2 + 6x_3 + x_4 + 7x_5 + 9x_6 + 10x_7 \\ & + 3x_8 + 11x_9 + 11x_{10} + 2x_{11} + x_{12} + 16x_{13} \\ & + 18x_{14} + 2x_{15} + x_{16} + x_{17} + 2x_{18} + 3x_{19} \\ & + 4x_{20} + 7x_{21} + 6x_{22} + 2x_{23} + 2x_{24} + x_{25} \\ & + 2x_{26} + x_{27} + 8x_{28} + 10x_{29} + 2x_{30} + x_{31} \\ & + 9x_{32} + x_{33} + 9x_{34} + 2x_{35} + 4x_{36} + 10x_{37} \\ & + 8x_{38} + 6x_{39} + x_{40} \le 25000, \\ & 5x_1 + 3x_2 + 2x_3 + 7x_4 + 7x_5 + 3x_6 + 6x_7 \\ & + 2x_8 + 15x_9 + 8x_{10} + 16x_{11} + x_{12} + 2x_{13} \\ & + 2x_{14} + 7x_{15} + 7x_{16} + 2x_{17} + 2x_{18} + 4x_{19} \\ & + 3x_{20} + 2x_{21} + 13x_{22} + 8x_{23} + 2x_{24} + 3x_{25} \\ & + 4x_{26} + 3x_{27} + 2x_{28} + x_{29} + 10x_{30} + 6x_{31} \\ & + 3x_{32} + 4x_{33} + x_{34} + 8x_{35} + 6x_{36} + 3x_{37} \\ & + 4x_{38} + 6x_{39} + 2x_{40} \le 25000, \\ & 3x_1 + 4x_2 + 6x_3 + 2x_4 + 2x_5 + 3x_6 + 7x_7 \\ & + 10x_8 + 3x_9 + 7x_{10} + 2x_{11} + 16x_{12} + 3x_{13} \\ & + 3x_{14} + 9x_{15} + 8x_{16} + 9x_{17} + 7x_{18} + 6x_{19} \\ & + 16x_{20} + 12x_{21} + x_{22} + 3x_{23} + 14x_{24} + 7x_{25} \\ & + 13x_{26} + 6x_{27} + 16x_{28} + 3x_{29} + 2x_{30} + x_{31} \\ & + 2x_{32} + 8x_{33} + 3x_{34} + 2x_{35} + 7x_{36} + x_{37} \\ & + 2x_{38} + 6x_{39} + 5x_{40} \le 25000, \\ & 10 \le x_i \le 99, i = 1,2,\cdots,20, \\ & 20 \le x_i \le 99, i = 21,22,\cdots,40 \end{aligned}$$

The best known function value given in [21] is $f(x^*) = 1030361$.

Problem 12[21][18]:

$$\begin{aligned} \min\ & f(x)=x_1^2+x_2^2+3x_3^2+4x_4^2+2x_5^2-8x_1-2x_2 \\ & -3x_3-x_4-2x_5 \\ \text{s.t.}\ & x_1+2x_2+2x_3+x_4+6x_5\le 800, \\ & 2x_1+x_2+6x_3\le 200, \\ & x_3+x_4+5x_5\le 200,\ x_1+x_2+x_3+x_4\ge 48, \\ & x_2+x_4+x_5\ge 34,\ 6x_1+7x_5\ge 104, \\ & 55\le x_1+x_2+x_3+x_4+x_5\le 400, \\ & 0\le x_i\le 99,\ x_i\in\mathscr{Z},\ i=1,2,3,4,5, \end{aligned}$$

The global minimum solution is $x^*=(16,22,5,5,7)^\top$ with $f(x^*)=807$.

Problem 13[21][18]:

$$\begin{aligned} \max\ & f(x)=x_1^2+x_1x_2-x_2^2+x_3x_1-x_3^2+8x_4^2 \\ & -17x_5^2+6x_6^3+x_4x_5x_6x_7+x_8^3+x_9^4-x_{10}^5 \\ & -x_{10}x_5+18x_3x_7x_6 \\ \text{s.t.}\ & 0\le x_i\le 99,\ x_i\in\mathscr{Z},\ i=1,2,\cdots,10 \end{aligned}$$

The global maximum solution is $x^*=(99,49,99,99,99,99,99,99,99,0)^\top$ with $f(x^*)=216300719$.

Problem 14[21][18]:

$$\begin{aligned} \min\ & f(u,v)=c^\top u-\tfrac{1}{2}u^\top Qu+d^\top v \\ \text{s.t.}\ & 2u_1+2u_2+v_6+v_7\le 10, \\ & 2u_1+2u_3+v_6+v_8\le 10, \\ & 2u_2+2u_3+v_7+v_8\le 10, \\ & -2u_4-v_1+v_6\le 0,\ -2v_2-v_3+v_7\le 0, \\ & -2v_4-v_5+v_8\le 0, \\ & -8u_i+v_{i+5}\le 0, i=1,2,3 \\ & u_j,v_k\in\{0,1\}, j=1,2,3,4, k=1,2,3,4,5,9 \\ & v_l\in\{0,1,2,3\}, l=6,7,8 \end{aligned}$$

where $c^\top=(5,5,5,5)$, $Q=10\cdot I_4$, $d^\top=(-1,\cdots,-1)$, $x=(u,v)$.
The global minimum solution is $x^*=(1,1,1,1,1,1,1,1,1,3,3,3,1)^\top$ with $f(x^*)=-15$.

Problem 15[21][18]:

$$\begin{aligned} \min\ & f(x)=\textstyle\sum_{i=1}^{9}\{\exp[-\frac{(u_i-x_2)^{x_3}}{x_1}]-\frac{i}{100}\}^2 \\ \text{s.t.}\ & 1\le x_1\le 100,\ 0\le x_2\le 25,\ x_i\in\mathscr{Z},\ i=1,2 \\ & x_3=j/2,\ 0\le j\le 10,\ j\in\mathscr{Z} \end{aligned}$$

where $u_i=25+[-50\log(i/100)]^{2/3}$.
The global minimum solution is $x^*=(50,25,1.5)^\top$ with $f(x^*)\approx 0$.

Problem 16[21]:

$$\begin{aligned} \min\ & f(x)=x^\top Qx \\ \text{s.t.}\ & \textstyle\sum_{i=1}^{n}\frac{x_i^2}{9n+i}\le 1,\ \sum_{i=1}^{n}ix_i\ge n/2 \\ & -5\le x_i\le 5,\ x_i\in\mathscr{Z},\ i=1,2,\cdots,n \end{aligned}$$

where $Q=[Q_{ij}]$, $Q_{ii}=2$, $Q_{ij}=1$ for $i\ne j$.
The global minimum solution is $x^*=(0,\cdots,0,0,-1,0,\cdots,0,1,0,\cdots,0)^\top$ with $f(x^*)=2$, where $x_i^*=-1$, $x_j^*=1$, $j\ge\lceil n/2\rceil+1$, $i\le j-\lceil n/2\rceil$ for all n.

Problem 17[21]:

$$\begin{aligned} \min f(x)=\ & 100(x_2-x_1^2)^2+(1-x_1)^2+90(x_4-x_3^2)^2 \\ & +(1-x_3)^2+10.1[(x_2-1)^2+(x_4-1)^2] \\ & +19.8(x_2-1)(x_4-1) \\ \text{s.t.}\ & -10\le x_i\le 10,\ x_i\in\mathscr{Z},\ i=1,2,3,4 \end{aligned}$$

The global minimum solution is $x^*=(1,1,1,1)^\top$ with $f(x^*)=0$.

Problem 18[21]:

$$\begin{aligned} \min\ & f(x)=[1.5-x_1(1-x_2)]^2+[2.25-x_1(1-x_2^2)]^2 \\ & +[2.625-x_1(1-x_2^3)]^2 \\ \text{s.t.}\ & x_i=0.001j,\ -10^4\le j\le 10^4,\ j\in\mathscr{Z},\ i=1,2 \end{aligned}$$

The global minimum solution is $x^*=(3,0.5)^\top$ with $f(x^*)=0$.

Problem 19[21]:

$$\begin{aligned} \min\ & f(x)=100(x_2-x_1^2)^2+(1-x_1)^2 \\ \text{s.t.}\ & x_1^2+x_2^2\ge 0.25,\ -\tfrac{1}{3}x_1+x_2\ge -0.1, \\ & x_i=j_i\times 10^{-4},\ 0\le j_i\le 10^5,\ j_i\in\mathscr{Z},\ i=1,2 \end{aligned}$$

The global minimum solution is $x^*=(1,1)^\top$ with $f(x^*)=0$.

Problem 20[21]:

$$\begin{aligned} \min\ & f(x)=(x_1+10x_2)^2+5(x_3-x_4)^2+(x_2-2x_3)^4 \\ & +10(x_1-x_4)^4 \\ \text{s.t.}\ & x_i=0.001j,\ -10^4\le j\le 10^4,\ j\in\mathscr{Z},\ i=1,2,3,4 \end{aligned}$$

The global minimum solution is $x^*=(0,0,0,0)^\top$ with $f(x^*)=0$.

Problem 21[21]:

$$\begin{aligned} \min\ & f(x)=g(x)h(x) \\ \text{s.t.}\ & x_i=0.001j,\ -2000\le j\le 2000,\ j\in\mathscr{Z},\ i=1,2 \end{aligned}$$

where $g(x)=1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2)$, $h(x)=30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)$.
The global minimum solution reported in [21] is $x^*=(0,-1)^\top$ with $f(x^*)=3$.

Problem 22[8]:

$$\begin{aligned} \min\ & f(x)=\frac{33.7539}{x_1}+\frac{1.4430}{x_2}+\frac{1.3885}{x_3} \\ \text{s.t.}\ & x_1+x_2+x_3=24 \\ & 1\le x_1\le 16, 1\le x_2\le 20, 1\le x_3\le 28, \\ & x_i\in\mathscr{Z},\ i=1,2,3 \end{aligned}$$

The global minimum solution is $x^*=(16,4,4)^\top$ with $f(x^*)=2.817494$.

References

[1] O. Hrstka, A. Kučerová, M. Lepš, and J. Zeman, Computers and Structures **81**, 1979-1990 (2003).

[2] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, IEEE Trans. Evol. Comput. **12**, 64-79 (2008).
[3] N. Noman and H. Iba, IEEE Trans. Evol. Comput. **12**, 107-125 (2008).
[4] S.Y. Ho, L.S. Shu, and J.H. Chen, IEEE Trans. Evol. Comput. **8**, 522-540 (2004).
[5] R. Ge, H. Huang, Applied Mathematics and Computation **34**, 39-60 (1989).
[6] W. Murray and K.M. Ng, Comput. Optim. Appl. **47**, 257-288 (2010).
[7] W.X. Zhu and M.M. Ali, Computer & Operations Research **36**, 2723-2728 (2009).
[8] W.X. Zhu and H. Fan, Journal of Computational and Applied Mathematics **223**, 356-373 (2009).
[9] N. Safaei, M. Saidi-Mehrabad, and M.S. Jabal-Ameli, European Journal of Operational Research **185**, 563-593 (2008).
[10] S.U. Seckiner and M. Kurt, Applied Mathematics and Computation **188**, 31-45 (2007).
[11] Y.C. Lin and K.S. Hwang, Computers and Mathematics with Applications **47**, 1295-1307 (2004).
[12] S. E. Kesen, S.K. Das and Z. Güngör, Computers & Operations Research **37**, 1148-1156 (2010).
[13] Z. Hua and F. Huang, Information Sciences **176**, 2869-2885 (2006).
[14] H. Li, Y.C. Jiao, L. Zhang, and Z.W. Gu, Proc. Advances in Natural Computation, Part 1, Lecture Notes in Computer Science 4221, Springer-Verlag, 696-705 (2006).
[15] F. Glover, Computers & Operations Research **33**, 2449-2494 (2006).
[16] K.E. Parsopoulos and M.N. Vrahatis, Natural Computing **1**, 235-306 (2002).
[17] W. N. Chen, J. Zhang, H. S. H. Chung, et al, IEEE Trans. Evol. Comput., **14**, 278-300 (2010).
[18] C. Mohan and H.T. Nguyen, Comput. Optim. Appl. **14**, 103-132 (1999).
[19] K.T. Fang and C.X. Ma, Orthogonal and Uniform Experimental Design (Science Press, Beijing, 2001).
[20] Y.W. Leung and Y. Wang, IEEE Trans. Evol. Comput. **5**, 41-53 (2001).
[21] C.K. Ng, L.S. Zhang, D. Li, and W.W. Tian, Comput. Optim. Appl. **31**, 87-115 (2005).
[22] C.K. Ng, D. Li, and L.S. Zhang, J. Glob. Optim. **37**, 357-379 (2007).
[23] T. P. Runarsson and X. Yao, IEEE Trans. Syst., Man, Cybern. C, Appl. Rev. **35**, 233-243 (2005).
[24] J.T. Tsai, T.K. Liu, and J.H. Chou, IEEE Trans. Evol. Comput. **8**, 365-377 (2004).
[25] Q. Zhang and Y.W. Leung, IEEE Trans. Evol. Comput. **3**, 53-62 (1999).

Hong Li received his Ph.D. degree in intelligent information processing from Xidian University in 2009. He is currently working as an associate professor in Xidian University. His research interests include evolutionary computation, optimization theory, algorithms, and applications.

Li Zhang received her Ph.D. degree in signal and information processing from Xidian University in 2012. She is currently working as an associate professor in Xidian University. Her research interests include neural network, intelligent information processing.

Yong-Chang Jiao received his Ph.D. degree from Xidian University in 1990. Currently, he is a professor with the Institute of Antennas and Electromagnetic Scattering, Xidian University. His research interests include optimization algorithms and applications, evolutionary computation, as well as analysis and synthesis of antennas.