Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

Swarm intelligence for mixed-variable design optimization*

GUO Chuang-xin (郭创新), HU Jia-sheng (胡家声), YE Bin (叶 彬), CAO Yi-jia (曹一家)†

(*College of Electrical Engineering, Zhejiang University, Hangzhou 310016, China*)

†E-mail: yijiacao@cee.zju.edu.cn

Received Oct. 9, 2003; revision accepted Nov. 29, 2003

Abstract: Many engineering optimization problems frequently encounter continuous variables and discrete variables which adds considerably to the solution complexity. Very few of the existing methods can yield a globally optimal solution when the objective functions are non-convex and non-differentiable. This paper presents a hybrid swarm intelligence approach (HSIA) for solving these nonlinear optimization problems which contain integer, discrete, zero-one and continuous variables. HSIA provides an improvement in global search reliability in a mixed-variable space and converges steadily to a good solution. An approach to handle various kinds of variables and constraints is discussed. Comparison testing of several examples of mixed-variable optimization problems in the literature showed that the proposed approach is superior to current methods for finding the best solution, in terms of both solution quality and algorithm robustness.

Key words: Swarm intelligence, Mixed variables, Global optimization, Engineering design optimization
Document code: A **CLC number:** TQ150.9; O646.5; X783

INTRODUCTION

Most nonlinear optimization methods assume that objective function variables are continuous. However, many practical engineering design problems frequently encounter discrete variables as well as continuous variables. Discrete variables are used in many ways such as the representation of the set of standard sized components, the decision on the number of identical parts or the choice between different design options. For example, the number of the teeth of a gear must be chosen as integers; whether or not to select a unit is a zero-one variable; the size of standard diametric pitch of a gear and the capacity of a power unit are discrete variables. The engineering optimal design which contains integer, discrete, zero-one and continuous variables is often referred to as a mixed-variable nonlinear optimization problem. Some reported methods for this class of problem are given in Table 1. Sandgren (1990) and Hajela and Shih (1989) have proposed nonlinear branch and bound algorithms, which were modified from the most widely used method in integer programming. These methods first obtained a solution by ignoring the discrete conditions. Cha and Mayne (1989) proposed an algorithm which combined a sequential quadratic programming (SQP) method or a local search strategy. The algorithm utilizes recursive quadratic programming and incorporates a Hessian matrix estimation. The algorithm was reported to be efficient based on the test problems, in terms of a low number of objective function and constraint evaluations. One deficiency was that when the objective function was non-convex, the direction for the local search guided by the downhill gradient vector may be wrong and the global optimum will not be reached. Loh and Papa-

*Project supported by the National Natural Science Foundation of China (Nos. 60074040, 6022506) and the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of China

Table 1 Alternative methods proposed to solve mixed-variable optimization problems

Method	Author (label)	Reference
Branch & bound using SQP	Sandgren (S90)	Sandgren (1990)
Recursive quadratic programming	Cha and Mayne (C89)	Cha and Mayne (1989)
Sequential linearization algorithm	Loh and Paplambros (L91)	Loh and Paplambros (1991a; 1991b)
Integer-discrete-continuous non-linear programming	Fu *et al.*(F91)	Fu *et al.*(1991)
Nonlinear mixed-discrete programming	Li and Chou (L94)	Li and Chou (1994)
Simulated annealing	Zhang and Wang (Z93)	Zhang and Wang (1993)
Genetic algorithm	Chen and Tsao (C93)	Chen and Tsao (1993)
Meta-genetic algorithm	Wu and Chow (W95)	Wu and Chow (1995)
Modified genetic algorithm	Lin *et al.*(L95)	Lin *et al.*(1995)
Evolutionary programming	Cao *et al.*(C97)	Cao *et al.*(1997)
Evolutionary strategy	Thierauf and Cai (T97)	Thierauf and Cai (1997)

lambros (1991a) introduced a sequential linearization approach for solving mixed-discrete nonlinear optimization problems and provided test results (Loh and Papalambros, 1991b). It was reported that this algorithm led to lower number of objective and constraint function evaluations than other major mixed nonlinear algorithms did. For non-convex problems, however, this algorithm might not find the global optimum since the linearization cut off part of the nonlinear feasible region. Fu *et al.*(1991) developed an interior penalty approach, which converted a constrained optimization problem into a sequence of unconstrained problems. The objective function for the unconstrained problem at each step of the sequential optimization included terms that introduce a penalty depending on the degree of constraint violation. Test results on five problems were provided to demonstrate the solution quality. However, like other penalization-based method, difficulties exist in selecting penalty function parameters and finding the global optimum in some cases. Modern heuristic algorithms were recently considered as effective tools for nonlinear optimization problems. These algorithms included simulated annealing, genetic algorithm, evolutionary programming, etc. (Cao and Wu, 1997; Cao *et al.*, 2000; Chen and Tsao, 1993; Fu *et al.*, 1991; Li and Chou, 1994; Lin *et al.*, 1995; Thierauf and Cai, 1997; Wu and Chow, 1995). These algorithms do not require that the objective function must be differentiable and continuous.

Among the above methods, no method appears to be substantially better than the others when applicability, efficiency, robustness and solution quality are all considered and very few of them can find a globally optimal solution when the objective functions are non-convex and non-differentiable. For these reasons, a reliable global approach to mixed-variable optimization would be of considerable value to both the optimization and the engineering design community.

In response to this demand, a novel approach for solving mixed-variable nonlinear optimization problem has been developed based on recently introduced swarm intelligence algorithms. Swarm intelligence, as demonstrated by natural biological swarms, exhibits numerous powerful features that are desirable in many engineering systems and can be applied to nonlinear and non-continuous optimization problems. This work was devoted to developing hybrid swarm intelligence approach (HSIA) for engineering optimization problems, which contain integer, discrete, zero-one and continuous variables. Implementation details of the approach to deal with various types of variables and constraints are discussed. Computational results of the examples on the mixed-variable optimization problems presented showed that HSIA is capable of finding the best solution of mixed-variable optimization problems.

MATHMATICAL FORMULATION OF THE MIXED-VARIABLE OPTIMIZATION PROBLEMS

In general, the main objective of mixed-variable design optimization is to determine the various design variables to minimize an objective function subjected to some design constraints. Let n_c, n_z, n_i and n_d be the number of continuous variables, zero-one variables, integer variables and discrete variables, respectively. The total number of variables is $n=n_c+n_z+n_i+n_d$. Then the mixed-variable optimization problem may be expressed in mathematical form as follows:

$$
\begin{aligned}
&\min f(\boldsymbol{X}) \qquad (1)\\
&\text{s.t. } h_i(\boldsymbol{X})=0 \quad i=1,2,\dots,m\\
&\quad g_i(\boldsymbol{X})\le 0 \quad i=m+1,\dots,p\\
&\boldsymbol{X}=\begin{pmatrix}\boldsymbol{X}^{\mathrm{c}}\\ \boldsymbol{X}^{\mathrm{z}}\\ \boldsymbol{X}^{\mathrm{i}}\\ \boldsymbol{X}^{\mathrm{d}}\end{pmatrix}\\
&\quad =\left[x_1^{\mathrm{c}},\dots,x_{n_{\mathrm{c}}}^{\mathrm{c}},x_1^{\mathrm{z}},\dots,x_{n_{\mathrm{z}}}^{\mathrm{z}},x_1^{\mathrm{i}},\dots,x_{n_{\mathrm{i}}}^{\mathrm{i}},x_1^{\mathrm{d}},\dots,x_{n_{\mathrm{d}}}^{\mathrm{d}}\right]^{\mathrm{T}}\\
&\quad x_i^{\mathrm{cl}}\le x_i^{\mathrm{c}}\le x_i^{\mathrm{cu}},\quad i=1,2,\dots,n_{\mathrm{c}}.\\
&\quad x_i^{\mathrm{zl}}\le x_i^{\mathrm{z}}\le x_i^{\mathrm{zu}},\quad i=1,2,\dots,n_{\mathrm{z}}.\\
&\quad x_i^{\mathrm{il}}\le x_i^{\mathrm{i}}\le x_i^{\mathrm{iu}},\quad i=1,2,\dots,n_{\mathrm{i}}.\\
&\quad x_i^{\mathrm{dl}}\le x_i^{\mathrm{d}}\le x_i^{\mathrm{du}},\quad i=1,2,\dots,n_{\mathrm{d}}.
\end{aligned}
$$

where $\boldsymbol{X}^{\mathrm{c}}\in\boldsymbol{R}^{\mathrm{c}}$, $\boldsymbol{X}^{\mathrm{z}}\in\boldsymbol{R}^{\mathrm{z}}$, $\boldsymbol{X}^{\mathrm{i}}\in\boldsymbol{R}^{\mathrm{i}}$ and $\boldsymbol{X}^{\mathrm{d}}\in\boldsymbol{R}^{\mathrm{d}}$ denote feasible subset of continuous, zero-one, integer and discrete variables, respectively. $x_i^{\mathrm{cl}}, x_i^{\mathrm{zl}}, x_i^{\mathrm{il}}$ and x_i^{dl} are the lower bounds of *i*th continuous variable, zero-one variable, integer variable and discrete variable, respectively. $x_i^{\mathrm{cu}}, x_i^{\mathrm{zu}}, x_i^{\mathrm{iu}}$ and x_i^{du} are the upper bounds of *i*th continuous variable, zero-one variable, integer variable and discrete variable, respectively. Here $x_i^{\mathrm{zl}}=0$ and $x_i^{\mathrm{zu}}=1$.

The above optimization problem is basically the same as general nonlinear programming except that the design variables may take on any form of continuous, zero-one, integer and discrete variable. Zero-one variables are usually required in the formulation of design problem with alternative option selections. Integer variables are used to formulate design problems with a number of objects, such as the number of gear teeth. Discrete variables are needed to represent standard parameters, for example, the diameter of a standard sized bolt and the capacity of a standard power unit. It had been shown that the presence of nonlinear discrete variables considerably adds to the solution complexity (Hajela and Shih, 1989; Cha and Mayne, 1989).

HYBRID SWARM INTELLIGENCE APROACH

Overview of swarm intelligence optimization

Swarm intelligence appears in biological swarms of certain insect species (Bonabeau *et al.*, 1999). It gives rise to complex and often intelligent behavior through complex interaction of thousands of autonomous swarm members. Interaction is based on primitive instincts with no supervision. The end result is accomplishment of very complex forms of social behavior and fulfillment of a number of optimization and other tasks. The main principle behind these interactions is called stigmergy, or communication through the environment. An example is pheromone laying on trails followed by ants. Pheromone is a potent form of hormone that can be sensed by ants as they travel along trails. It attracts ants and therefore ants tend to follow trails that have high pheromone concentrations. This causes an autocatalytic reaction, i.e., one that is accelerated by itself. Ants attracted by the pheromone will lay more of the same on the same trail, causing even more ants to be attracted. This characteristic makes swarm intelligence very attractive for routing networks, robotics and optimization (Dorigo *et al.*, 1996; Wang *et al.*, 2002). Moreover, Dorigo *et al.*(1996) developed Ant Colony Optimization (ACO) mainly based on the social insect, especially ant, behavior. They called the research field as swarm intelligence. Namely, they showed that the swarm behavior could be utilized as an optimization procedure.

Simulation of a bird swarm was used to develop a particle swarm optimization (PSO) concept (Kennedy and Eberhart, 1995; Kennedy *et al.*, 2001). PSO was basically developed through simulation of bird flocking in two-dimension space. According to the research results for a flock of birds, birds find food by flocking (not by each individual). The observation leads to the assumption that every information is shared inside flocking. Moreover, according to observation of behavior of human groups, behavior of each individual (agent) is also based on behavior patterns authorized by the groups such as customs and other behavior patterns according to the experiences of each individual. The position of each agent is represented by *xy*-axis position and the velocity (displacement vector) is expressed by v_x (the velocity of *x*-axis) and v_y (the velocity of *y*-axis). Modification of the agent position is realized by using the position and the velocity information.

Searching procedures by PSO based on the above concept can be described as follows: a flock of agents optimizes a certain objective function. Each agent knows its best value so far (*pbest*) and its *xy* position. The information corresponds to the personal experiences of each agent. Moreover, each agent knows the best value so far in the group (*gbest*) among *pbests*. The information corresponds to knowledge of how the other agents around them have performed. Namely, each agent tries to modify its position using the following information:

- the distance between the current position and *pbest*,
- the distance between the current position and *gbest*.

This modification can be represented by the concept of velocity. Velocity of each agent can be modified by the following equation:

$$v_i^{k+1} = w_i v_i^k + c_i rand_1 \times (pbest_i - s_i^k) + c_2 rand_2 \times (gbest - s_i^k) \quad (2)$$

where v_i^k is current velocity of agent *i* at iteration *k*, $rand_1$ and $rand_2$ are random numbers between 0 and 1, s_i^k is current position of agent *i* at iteration *k*, $pbest_i$ is *pbest* of agent *i*, *gbest* is *gbest* of the group, w_i is weight function for velocity of agent *i*, c_i is weight coefficients for each term.

The right-hand-side (RHS) of Eq.(2) consists of three terms. The first term is the previous velocity of the agent. The second and third terms are utilized to change the velocity of the agent. Without the second and third terms, the agent will keep on 'flying' in the same direction until it hits the boundary. Namely, it tries to explore new areas and, therefore, the first term corresponds to diversification in the search procedure. On the other hand, without the first term, the velocity of the 'flying' agent is only determined by using its current position and its best positions in history. Namely, the agents will try to converge to the their *pbests* and/or *gbest* and, therefore, the terms correspond to intensification in the search procedure.

The following weighting function is usually used in Eq.(2):

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \quad (3)$$

where $w_{\max}$ is the initial weight, $w_{\min}$ is final weight, $iter_{\max}$ is maximum iteration number, *iter* is current iteration number. Using the above Eq.(3), diversification characteristics is gradually decreased.

Using the above Eq.(2), a certain velocity that gradually gets close to *pbests* and *gbest* can be calculated. The current position (searching point in the solution space) can be modified by the following equation:

$$s_i^{k+1} = s_i^k + v_i^{k+1} \quad (4)$$

Fig.1 shows the above concept of modification of searching points.

The features of the searching procedure can be summarized as follows:

- PSO utilizes several searching points like genetic algorithm and the searching points gradually get close to the optimal point using their *pbests* and the *gbest*.
- The first term of RHS of Eq.(2) corresponds

to diversification in the search procedure. The second and third terms of that correspond to intensification in the search procedure. Namely, the method has a well-balanced mechanism to utilize diversification and intensification in the search procedure efficiently.

• The original PSO can be applied to the only continuous problem. However, the method can also be expanded to the discrete problem using discrete numbers like grids for *xy* position and its velocity.

• The above concept is explained using only *xy*-axis (two-dimension space). However, the method can be easily applied to *n*-dimension problem.

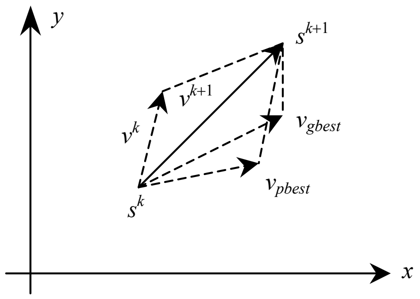

Fig.1 Modification process of the swarm intelligence approach

Treatment of constraints and discrete state variables

Most engineering design optimization problems are constrained. Two techniques are often used to handle constraints. One is a variable restriction method which restricts the solution space to the solutions which conform to the constraints. The other is a penalty function method which allows solutions which violate the constraints at the expense of a suitably defined penalty function. However, determining appropriate penalty coefficients is not an easy task, it must be estimated based on the relative scaling of the distance metrics of multiple constraints, the difficulty of satisfying a constraint, and the seriousness of a constraint violation, or be determined experimentally. Although the adaptive penalty strategies proved to be effective in some cases, they are still quite probelm-dependent (Coit *et al*., 1996). It was also noticed that if the values of the objective function were very large, then small differences between objective values were not easily identified, and this was undesirable, a leading cause of premature convergence in evolutionary computation.

In this study, the variable restriction method was used, in which a modification based on Eqs.(2) and (4) was first randomly done, and the solution was then checked against the constraints. If the constraints were satisfied, the modification was usable, otherwise, the modification was discarded and a new modification was generated and checked. It should be noted that some of the modifications generated were unusable, and therefore more computational time might be needed.

The state variables were treated in PSO as follows: for continuous variables, initial values were generated randomly between upper and lower bounds of the specification values. The value was also modified in the search procedure between the bounds. For discrete variables, they could be handled in Eqs.(2) and (3) with little modification, i.e., as though they were continuous with nearest available discrete values then being chosen. If a discrete random number was used in Eq.(2) and the whole calculation of the right-hand side (RHS) of Eq.(2) was discretized to the existing discrete number, both continuous and discrete number could be handled in the algorithm with no inconsistency.

Hybrid swarm intelligence approach

Using the above concepts, the whole HSIA algorithm can be expressed as follows:

Step 1. State variables (searching point)

State variables (states and their velocities) can be expressed as vectors of continuous and discrete numbers. HSIA utilizes multiple searching points for search procedures.

Step 2. Generation of initial searching points

Initial searching points and velocities of agents are generated using the above-mentioned state variables randomly.

Step 3. Evaluation of searching points

The current searching points are evaluated by using the objective functions of the target problem. *pbests* and *gbest* can be modified by comparing the evaluation values of the current searching points, *pbests* and *gbest*.

Step 4. Modification of searching points

The current searching points are modified using the Eqs.(2), (3) and (4).

Step 5. Stop criterion

The search procedure can be stopped when the current iteration number reaches the predetermined maximum iteration number. The last *gbest* can be output as a solution.

COMPUTATION EXAMPLES

To demonstrate the effectiveness of the proposed approach, three numerical examples, i.e. design of a gear train, design of a pressure vessel, and design of a coil spring, were optimized using HSIA. These non-linear, engineering design problems with discrete, integer and continuous variables were first investigated by many other researchers (see Table 1 for details). These problems represent optimization situations involving discrete, integer and continuous variables that are similar to those encountered in everyday mechanical engineering design tasks. As the problems are clearly defined and fairly easy to understand, they form a suitable basis for comparing alternative optimization methods.

The HSIA algorithm has been implemented using the C++ language. In the algorithm, the population size, the weighting coefficients w_{max}, w_{min}, and control parameters c_1 and c_2 were chosen as 30, 0.9, 0.4, 1.496 and 1.496 respectively. The performance of the developed HSIA algorithm was tested on three typical mechanical design optimization problems: gear train design, pressure vessel design and coil spring design.

Example 1: Design of a gear train

The first example problem was to optimize the gear ratio for the compound gear train arrangement shown in Fig.2. The gear ratio for a reduction gear train is defined as the ratio of the angular velocity of the output shaft to that of the input shaft. In order to produce the desired overall gear ratio, the compound gear train is constructed out of two pairs of the gearwheels, d–a and b–f. The overall gear ratio i_{tot} between the input and output shafts can be expressed as:

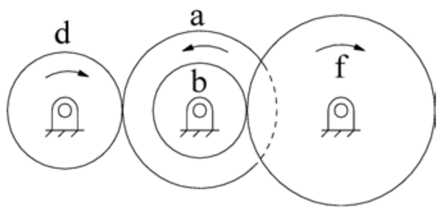

Fig.2 Gear train design

$$i_{tot} = \frac{w_o}{w_i} = \frac{z_d z_b}{z_a z_f} \quad (5)$$

where w_o and w_i are the angular velocities of the output and input shafts, respectively, and z denotes the number of teeth on each gearwheel. It is desirable to produce a gear ratio as close as possible to 1/6.931. For each gear, the number of teeth must be from 12 to 60. The design variables are denoted by a vector $\boldsymbol{X}=[x_1, x_2, x_3, x_4]^T=[z_d, z_b, z_a, z_f]^T$, and hence x_1, x_2, x_3 and x_4 are the numbers of teeth of gears d, b, a and f, respectively, which must be integers.

The optimization problem is expressed as:

$$\min f = \left(\frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4}\right)^2 \quad (6)$$

$$\text{s.t.} \quad 12 \le x_i \le 60 \quad i = 1,...,4$$

The gear train problem was solved using the HSIA approach and Table 2 lists the various gear train solutions and compares HSIA's result with those reported by many other researchers. We can observe from Table 2 that the solution found by HSIA was equally as good as the best solution in the literature. In addition, it should be noted that HSIA provided different results from run to run with the same objective function value (Table 3). In fact, by inspecting Eq.(6), it is obvious that there are four global optima. Because HSIA can work with a population of solutions rather than just a single solution, it is capable of finding multiple global optima for this problem. By using a sufficiently large population, it is possible to obtain all four alternative solutions in a single run. Despite that, only one solution was extracted from the population of the last generation because the other three solutions can be found in a trivial way based on one solution and the symmetry of Eq.(6). In practice, however, there exist optimization tasks with multiple global optima that cannot be detected so simply.

Example 2: Design of a pressure vessel

The second example is to design a compressed air storage tank with a working pressure of 3000 psi and a minimum volume of 750 ft^3. The schematic of a pressure vessel is shown in Fig.3. The cylindrical pressure vessel is capped at both ends by hemispherical heads. Using rolled steel plate, the shell is to be made in two halves that are joined by two longitudinal welds to form a cylinder. Each head is forged and then welded to the shell. Let the design variables be denoted by the vector $\boldsymbol{X}=[x_1, x_2, x_3, x_4]^{\mathrm{T}}$, x_1 is the spherical head thickness, x_2 is the shell thickness, x_3 and x_4 are the radius and length of the shell, respectively. The objective in this example is to minimize the manufacturing cost of the pressure vessel. The manufacturing cost of the pressure vessel is a combination of material cost, welding cost and forming cost. Readers can refer to Sandgren (1990) for more details on how cost is determined. The constraints are set in accordance with respective ASME codes. The mathematical model of the problem is:

$$\begin{aligned}\min\ & f(\boldsymbol{X}) = 0.6224x_1x_2x_3x_4 + 1.7781x_2x_3^2 \\ & \qquad + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \qquad (7)\\ \text{s.t.}\ & g_1(\boldsymbol{X}) = 0.0193x_3 - x_1 \le 0 \\ & g_2(\boldsymbol{X}) = 0.00954x_3 - x_2 \le 0 \\ & g_3(\boldsymbol{X}) = 750\times1728 - \pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 \le 0 \\ & g_4(\boldsymbol{X}) = x_4 - 240 \le 0\end{aligned}$$

where the design variables, x_3 and x_4, are continuous, while x_1 and x_2 are discrete values and integer multiplies of 0.0625 inch respectively, in accordance with the available thickness of rolled steel plates.

The values of the variables, constraints and objective function correspond to the optimum solution of this problem obtained by employing the HSIA algorithm are listed in Table 4. It can be seen from the table that the HSIA approach has found a

Table 2 Optimal solutions for the gear train design

Item	S90	L91	F91	Z93	L95	C97	HSIA
x_1	18	19	14	30	19	30	16
x_2	22	16	29	15	16	15	19
x_3	45	42	47	52	43	52	43
x_4	60	50	59	60	49	60	49
$f(\boldsymbol{X})$	5.7×10^{-6}	0.233×10^{-6}	4.5×10^{-6}	2.36×10^{-9}	2.7×10^{-12}	2.36×10^{-9}	2.7×10^{-12}
Gear ratio	0.146666	0.144762	0.146411	0.144231	0.144281	0.144231	0.144281
Error (%)	1.65	0.334	1.17	0.033	0.0011	0.033	0.0011

Table 3 Alternative solutions for the gear train design found by HSIA

Solution	z_d	z_b	z_a	z_f
1	19	16	43	49
2	16	19	43	49
3	19	16	49	43
4	16	19	49	43

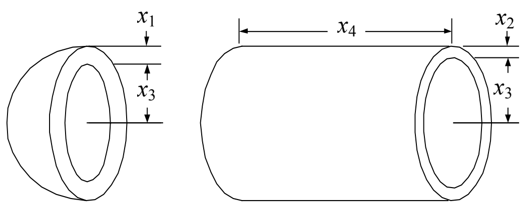

Fig.3 Pressure vessel design

better solution than that ever reported before. It should also be pointed out that for this problem, some researchers obtained significantly lower objective function values (Cao and Wu, 1997; Li and Chou, 1994; Thierauf and Cai, 1997). However, all of the solutions lie in the modification of constraints via the extending region of the search space. Because of that, they could not be fairly compared with the results obtained using Sandgren's original problem (Sandgren, 1990).

Example 3: Design of a coil compression spring

The spring is to be a helical compression spring shown in Fig.4. A strictly axial and constant load is to be applied to the spring. The objective is to minimize the volume of spring steel wire used to manufacture the spring (minimum weight). The design variables are the number of spring coils, N, the outside diameter of the spring, D, and the spring wire diameter, d. This example contains integer, discrete and continuous variables. The number of spring coils, N, is an integer variable and the outside diameter, D, is a continuous variable. The spring wire diameter, d, may have only discrete values according to available standard spring steel wire diameters shown in Table 5. Let $\boldsymbol{X}=[x_1, x_2, x_3]=[N, D, d]$, the problem is formulated as follows:

$$\min f(\boldsymbol{X}) = \frac{\pi x_2 x_3^2 (x_1+2)}{4} \tag{8}$$

$$\text{s.t.} \quad g_1(\boldsymbol{X}) = \frac{8C_f F_{\max} x_2}{\pi x_3^2} - S \le 0$$

$$g_2(\boldsymbol{X}) = l_f - l_{\max} \le 0$$

$$g_3(\boldsymbol{X}) = d_{\min} - x_3 \le 0$$

$$g_4(\boldsymbol{X}) = x_2 - D_{\max} \le 0$$

$$g_5(\boldsymbol{X}) = 3.0 - \frac{x_2}{x_3} \le 0$$

$$g_6(\boldsymbol{X}) = \sigma_p - \sigma_{pm} \le 0$$

$$g_7(\boldsymbol{X}) = \sigma_p + \frac{F_{\max} - F_p}{K} + 1.05(x_1+2)x_3 - l_f \le 0$$

$$g_8(\boldsymbol{X}) = \sigma_w - \frac{F_{\max} - F_p}{K} \le 0$$

where

$$C_f(\boldsymbol{X}) = \frac{4(x_2/x_3)-1}{4(x_2/x_3)-4} + \frac{0.615x_3}{x_2} \le 0$$

$$K = \frac{Gx_3^4}{8x_1x_2^3}$$

Table 4 Optimal solutions for the pressure vessel design

item	S90	F91	W95	HSIA
x_1	1.125	1.125	1.125	1.125
x_2	0.625	0.625	0.625	0.625
x_3	48.67	48.38	58.19	58.29
x_4	106.72	111.75	44.29	43.70
$-g_1(\boldsymbol{X})$	0.179	0.191	0.001782	3.0×10^{-6}
$-g_2(\boldsymbol{X})$	0.1578	0.163	0.06979	0.0689
$-g_3(\boldsymbol{X})$	3.0	75.875	974.58	69.24
$-g_4(\boldsymbol{X})$	133.28	128.255	195.70	196.30
$f(\boldsymbol{X})$	7982.5	8048.62	7207.50	7197.9

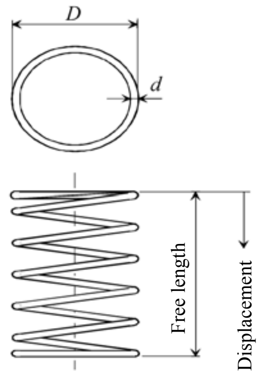

Fig.4 Design of a coil compression spring

Table 5 Allowable spring steel wire diameters (inch)

0.009	0.0095	0.0104	0.0118	0.0128	0.0132	0.014
0.015	0.0162	0.0173	0.018	0.020	0.023	0.025
0.028	0.032	0.035	0.041	0.047	0.054	0.063
0.072	0.080	0.092	0.207	0.225	0.244	0.263
0.162	0.177	0.192	0.207	0.225	0.244	0.263
0.283	0.307	0.331	0.362	0.394	0.4375	0.500

$$\sigma_{\mathrm{p}} = \frac{F_p}{K}$$

$$l_f = \frac{F_{\max}}{K} + 1.05(x_1 + 2)x_3$$

The objective function $f(\boldsymbol{X})$ computes the volume of spring steel wire as a function of the design variables. The detailed explanation about the design constraints and values for the parameters can be found in (Sandgren, 1990).

The solution for coil spring problem was compared with the results obtained by other researchers in Table 6. As shown, HSIA obtained a better solution for the coil spring design problem than the best solution found in literature. In order to demonstrate the robustness of the optimum seeking algorithm, 100 independent runs of the optimization program were performed. All performed runs of HSIA yielded the reported value of $f(\boldsymbol{X})$. All of the solutions were within the feasible region of the design space, too. It is sometimes argued that the evolutionary optimization requires a large number of cost function evaluations. Thus, the computational cost would be high. Before deciding whether or not this is true in case of HSIA, some further facts should be considered. The real-world optimization problem discussed here can be solved on an ordinary PC (2.4 GHz Pentium IV) using less than 1 second of computation time.

CONCLUSIONS AND FUTURE WORK

In this paper, a hybrid swarm intelligence algorithm was developed for engineering design optimization problems containing integer, discrete, zero-one and continuous variables. The approach to the diverse variable representations and constraints was discussed. Some test examples were given to show that the developed HSIA algorithm was superior to the existing methods of this kind for finding the best solution, both in the solution quality and algorithm robustness. Future research should concentrate mainly on developing self-adaptation techniques for the modification control parameters and penalty functions to reduce the number of unusable modifications and improve the solution quality.

References

Bonabeau, E., Dorigo, M., Theraulaz, G., 1999. Swarm Intelligence: from Natural to Artificial Systems. Oxford University Press.

Cao, Y.J., Wu, Q.H., 1997. Mechanical Design Optimization by Mixed-variable Evolutionary Programming. Proceedings of the 1997 IEEE Conference on Evoluionary Computation, IEEE Press, p.443-446.

Cao, Y.J., Jiang, L., Wu, Q.H., 2000. An evolutionary programming approach to mixed-variable optimization problems. *International Journal of Applied Mathematical Modelling*, **24**(10):931-942.

Cha, J., Mayne, R., 1989. Optimization with discrete varia-

Table 6 Optimal solutions for the coil spring design

Item	S90	F91	W95	HSIA
x_1	10	9	9	9
x_2	1.180701	1.2287	1.2274	1.223
x_3	0.283	0.283	0.283	0.283
$-g_1(\boldsymbol{X})$	543.09	415.969	550.993	1008.81
$-g_2(\boldsymbol{X})$	8.8187	8.9207	8.9264	8.946
$-g_3(\boldsymbol{X})$	0.08298	0.083	0.083	0.083
$-g_4(\boldsymbol{X})$	1.1893	1.7713	1.7726	1.77696
$-g_5(\boldsymbol{X})$	1.1723	1.3417	1.3371	1.32170
$-g_6(\boldsymbol{X})$	5.4643	5.4568	5.4585	5.4643
$-g_7(\boldsymbol{X})$	0.0	0.0	0.0	0.0
$-g_8(\boldsymbol{X})$	0.0	0.0174	0.0134	0.0
$f(\boldsymbol{X})$	2.7995	2.6709	2.6681	2.659

bles via recursive quadratic programming: part II. *Transaction of the ASME*, **111**:130-136.

Chen, J.L., Tsao, Y.C., 1993. Optimal design of machine elements using genetic algorithms. *Journal of the Chinese Society of Mechanical Engineers*, **14**:193-199.

Coit, D.W., Smith, A.E., Tate, D.M., 1996. Adaptive penalty methods for genetic optimization of constrained combinatorial problems. *INFORMS J. Computing*, **8**: 173-182.

Dorigo, M., Maniezzo, V., Colorni, A., 1996. The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, **26**:29-41.

Fu, J.F., Fenton, R.G., Cleghorn, W.L., 1991. A mixed ineger-discrete-continuous programming method and its application to engineering design optimization. *Engineering Optimization*, **17**:263-280.

Hajela, P., Shih, C., 1989. Multiobjective optimum design in mixed-integer and discrete design variable problems. *AIAA Journal*, **28**:670-675.

Kennedy, J., Eberhart, R., 1995. Particle Swarm Optimization. Proceedings of IEEE International Conference on Neural Networks, p.1942-1948.

Kennedy, J., Eberhart, R., Shi., Y., 2001, Swarm Intelligence. Morgan Kaufmann Publishers, San Francisco.

Li, H.L., Chou, C.T., 1994. A global approach of nonlinear mixed discrete programming in design optimization. *Engineering Optimization*, **22**:109-122.

Lin, S.S., Zhang, C., Wang, H.P., 1995. On mixed-discrete nonlinear optimization problems: A comparative study. *Engineering Optimization*, **23**:287-300.

Loh, H.T., Papalambros, P.Y., 1991a. A sequential linearization approach for solving mixed-discrete nonlinear design optimization problems. *ASME Journal of Mechanical Design*, **113**:325-334.

Loh, H.T., Papalambros, P.Y., 1991b. Computational implementations and tests of a sequential linearization algorithm for mixed-discrete nonlinear design optimization problems. *ASME Journal of Mechanical Design*, **113**:335-345.

Sandgren, E., 1990. Nonlinear integer and discrete programming in mechanical design optimization. *ASME Journal of Mechanical Design*, **112**:223-229.

Thierauf, G., Cai, J., 1997. Evolution Strategies-parallelization ad Application in Engineering Optimization. *In*: B.H.V. Topping (ed.), Parallel and Distributed Processing for Computational Mechanics. SaxeCoburg Publications, Edinburgh.

Wang, H.F., Li, H., Chen, H., 2002. Power system voltage control by multiple STATCOMs based on learning humoral immune response. *IEEE Proc. Part C*, **149**: 301-305.

Wu, S.J., Chow, P.T., 1995. Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization. *Engineering Optimization*, **24**:137-159.

Zhang, C., Wang, H.P., 1993. Mixed-discrete nonlinear optimization with simulated annealing. *Engineering Optimization*, **21**:277-291.