Iterative Dynamic Diversity Evolutionary Algorithm for Constrained Optimization

GAO Wei-Shang[1, 2] SHAO Cheng[1, 2]

Abstract Evolutionary algorithms (EAs) were shown to be effective for complex constrained optimization problems. However, inflexible exploration in general EAs would lead to losing the global optimum nearby the ill-convergence regions. In this paper, we propose an iterative dynamic diversity evolutionary algorithm (IDDEA) with contractive subregions guiding exploitation through local extrema to the global optimum in suitable steps. In IDDEA, a novel optimum estimation strategy with multi-agents evolving diversely is suggested to efficiently compute dominance trend and establish a subregion. In addition, a subregion converging iteration is designed to redistrict a smaller subregion in current subregion for next iteration, which is based on a special dominance estimation scheme. Meanwhile, an infimum penalty function is embedded into IDDEA to judge agents and penalize adaptively the unfeasible agents with the lowest fitness of feasible agents. Furthermore, several engineering design optimization problems taken from the specialized literature are successfully solved by the present algorithm with high reliable solutions.

Key words Constrained optimization, evolutionary algorithm, multi-agents, swarm intelligence

Citation Gao Wei-Shang, Shao Cheng. Iterative dynamic diversity evolutionary algorithm for constrained optimization. *Acta Automatica Sinica*, 2014, **40**(11): 2469−2479

DOI 10.3724/SP.J.1004.2014.02469

A great deal of engineering design problems can be formulated as constrained optimization problems that often consist of many mixed equality and inequality constrains. These problems are difficult to solve because of their multi-constraint feature. Penalty functions are usually used to handle these multiple constraints[1−2], which will transform the problems into unconstrained ones but meanwhile make original objective function more complex. Previous researches[3−5] have suggested that evolutionary algorithms (EAs) can be widely used to tackle such problems. Many successful applications of EAs have been reported to solve engineering problems such as industrial design[6−7] and military management[8]. EAs can also be used to deal with complex optimization of regression[9] and classification[10] in machine learning. To extend the application of EAs to more difficult but important problems, Vural et al. carried out intensive study on analog filter design with evolutionary algorithms[11], Li et al. presented a new cooperatively coevolving particle swarm for large-scale optimization[12], Blackwell provided further study of collapse in bare bones particle swarm optimization[13], Pehlivanoglu enhanced particle swarm optimization with a periodic mutation strategy and neural networks[14], Chen et al. proposed particle swarm optimization with an aging leader and challengers[15], and Naznin et al. suggested a progressive alignment method using genetic algorithm for multiple sequence alignment[16]. Further analysis of particle swarm optimization algorithm was also carried out by Pan et al.[17] and Liu et al.[18]. Constrained optimization is an important kind of problems solved by EAs, such as the methods proposed by Wang et al.[19−20], Cai et al.[21], Krohling et al.[22] and Tessema et al.[23]. Recently Daneshyari et al.[24] have noted that genetic-based algorithms and swarm-based paradigms are two popular population-based heuristics introduced as EAs for solving constrained optimization problems[25−27]. All these algorithms have better global search capabilities by sharing the principle of natural evolution or swarm intelligence in nature.

Genetic algorithm (GA) is a search heuristic which mimics the process of natural evolution by parallel computing. It became popular through the work of John Holland in the early 1970s, and particularly his book *Adaptation in Natural and Artificial Systems* (1975). The strong exploration of GA is suitable for intricate optimization problems[28−29], but GA′s intrinsic disadvantages such as slow convergence caused by mutation operator[5] have limited the promotion of this algorithm in practical application. Furthermore, particle swarm optimization (PSO) developed by Kennedy and Eberhart[30−31] has received more and more attention regarding its potential as a faster global optimization technique[5]. However, it might be caught in the trap of local optimum caused by premature convergence. Thus, special trade-off between exploration and exploitation is required to balance optimization reliability and convergence speed. A considerable method is to form a hybrid algorithm with strong exploration and exploitation by combining a variety of EAs. It is because that mutually reinforcing will make hybrid algorithm fit for difficult optimization problems where acceptable solutions can be achieved[32−34]. Recently, co-evolutionary algorithms (CoEAs) have been extensively studied in solving complex constrained optimization problems[3]. It can be considered as a new form of hybrid algorithm with high efficiency in exchanging information between agents. Another novel EA, rain forest algorithm (RFA)[35], suggested by Gao et al. gets better features both in efficiency and accuracy. Although these algorithms can perform better than the standard EAs, inflexible exploration in general EAs would lead to losing the global optimum nearby the ill-convergence regions.

Usually the fitness function, especially around the global optimum, becomes much more complex because of the mixed constraints. Ill-convergence towards a region nearby the global optimum is likely to arise when dealing with such constrained optimization. Then inflexible exploration, such as mutation and scattering, will distribute other agents far from the local extremum and probably lose the global optimum. In this paper, we propose contractive subregions with flexible exploration guiding exploitation through local

Manuscript received June 17, 2013; accepted November 26, 2013
Supported by National Natural Science Foundation of China (61074020)
Recommended by Associate Editor HOU Zeng-Guang
1. School of Control Science and Engineering, Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China 2. Institute of Advanced Control Technology, Dalian University of Technology, Dalian 116024, China

extrema to the global optimum in suitable steps. An appropriate redistricting method is suggested to establish a smaller feasible region for regulating the exploration and enhancing the exploitation. By subregions, an iterative dynamic diversity evolutionary algorithm (IDDEA) with multi-agents is also devised to solve constrained optimization problems consisting of complex mixed equality and inequality constraints. A dynamic diversity evolutionary algorithm (DDEA) with high exploration ability is called in each iteration of IDDEA. Thus, personal best agents in each iteration of IDDEA can confirm a subregion in which the global optimization can be obtained. Once a subregion is found, another specific exploration and exploitation with faster and more accurate convergence will be carried out in the subregion. In addition, an effective and adaptive penalty function method that is fit for IDDEA is also illuminated in this paper. Unlike other penalty methods, this penalty term is not added to the objective function directly but constitutes the IDDEA as a correction term. The drawback of common penalty function methods is that they often require several parameters which are used to adjust the relative weights of each constraint and the weight of the penalty against the objective function. To avoid the difficulty in designing those parameters, we selected the worst agent as a reference term and added an infimum penalty function into IDDEA. The corresponding penalty function will be called whenever infeasible agents appear and reduce the fitness of these agents to a level just under the worst feasible agent. As infeasible solutions are banned by the constraints equally, the fitness of infeasible agents will also be modified according to the constraints equally.

This paper is organized as follows. Section 1 explains how to divide a proper subregion with DDES and presents an optimum estimation strategy. The iteration of IDDEA is suggested in Section 2 which illuminates a process to make the subregion contracted. An infimum penalty function and the process of IDDEA are proposed in Section 3. To confirm the capability of IDDEA, Section 4 and Section 5 provide several numerical experimentations across benchmark problems and comparisons with state-of-the-art methods. Conclusions and suggestions for further work are presented in Section 6.

1 How to divide a proper subregion

To divide a subregion containing the optimum in global area, an optimum estimation strategy and a subregion establishment method are proposed in this section. First of all, dynamic diversity evolutionary searching (DDES) with multi-agents is introduced to compute the dominance trend of an objective function. This searching process with strong exploration-exploitation ability will pick up the character of extremum distribution. Secondly, a subregion will be established according to the distribution of the global best agent and the personal best agents found by DDES.

1.1 Optimum estimation

The DDES with strong exploration-exploitation ability is well-suited for optimum estimation. Three types of agents, partition agents (PAs), basic agents (BAs) and creative agents (CAs), are combined together in DDES to realize a quick-shift between exploration and exploitation. Partition agents are distributed in feasible region uniformly and their gradual increase plays the role of brute search and ensures exploration throughout the course of optimization. Basic agents are distributed in different partitions according to the property of partition agents, which plays the role of transition media between partition agents and creative agents and also between exploration and exploitation. Creative agents are distributed around basic agents according to the property of basic agents, playing a role of exploitation in global area and also exploration in local partitions. The number of creative agents is the largest, which is fit for exploiting in many smaller but more advantageous areas. The cooperation of these agents is managed by DDES following a form of three layers proxy model (TLPM) as shown in Fig. 1.

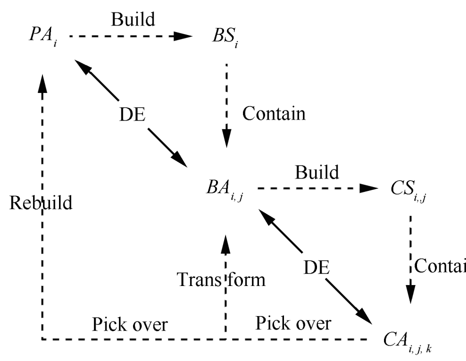

Fig. 1 Three layers proxy mode of DDES

The evolutionary process of DDEA is mainly carried out by the rebirth of new agents around the senior ones. BAs alternate with CAs, which are also affected by PAs. The focus in DDEA is no longer on the position updating of previous agents but on the density distribution of newborn agents. The density of newborn agents is controlled by regulating the range and the scale of newborn agents according to the feedback of sampling information. $\widehat{AD}$, $\widehat{RD}$ and $\widehat{DC}$ are introduced in this paper as correction factors to analyze the sampling information and estimate the situation of each agent, which will guide newborn agents to distribute with adaptive density. The correction factors can be defined as below:

$$\begin{aligned}
\widehat{AD} =& \exp\left(\frac{F-\frac{F_{\max}+F_{\min}}{2}}{\frac{F_{\max}-F_{\min}}{2}}\right) = \\
& \exp\left(\frac{2F-F_{\max}-F_{\min}}{F_{\max}-F_{\min}}\right) \\
\widehat{RD} =& \begin{cases} \exp\left(\frac{G-\frac{G_{\max}}{2}}{\frac{G_{\max}}{2}}\right), & \text{if } G>0 \\ 0, & \text{if } G\le 0 \end{cases} = \\
& \begin{cases} \exp\left(\frac{2G}{G_{\max}}-1\right), & \text{if } G>0 \\ 0, & \text{if } G\le 0 \end{cases} \\
\widehat{DC} =& (1-\gamma)\times\widehat{RD}+\gamma\times\widehat{AD}
\end{aligned}$$

where F refers to the fitness of an agent, $F_{\max}$ and $F_{\min}$ refer to the minimum and the maximum fitness in a region, respectively. G refers to the growth of fitness found by an agent in one iteration, $G_{\max}$ refers to the largest G found by all the agents in one iteration, and γ is a weight number that will increase gradually from zero to one in the process of optimization.

$\widehat{AD}$ plays an important role in comparing the fitness of newborn agents by mapping their fitness into an interval of $[1/e, e]$ in the form of increasing function. It indicates the situation of current agent′s environment and will be used to regulate the range of newborn agents in next iteration. $\widehat{RD}$ plays an important role in comparing the fitness growth of newborn agents by mapping their fitness growth into an interval of $[1/e, e]$ in the form of increasing function. It indicates the complexity of current agent′s environment and will be used to regulate the scale of newborn agents in next iteration. $\widehat{DC}$ represents a transition from $\widehat{RD}$ to $\widehat{AD}$.

The range of newborn agents is determined by two items. The first one is narrowing range with $\frac{2}{3}$ ratio, which means the child agents will inherit $\frac{2}{3}(r_{parent})$ from their parents. The second one is regulated range with $\widehat{AD}$, which means the child agents will study from the sampling information and adjust their range toward $\frac{2}{3}(\frac{r_{parent}}{\widehat{AD}})$. Then, the range of those agents whose fitness is lower will be set to a larger value and vice versa. To make information fusion, α is introduced as study factor to integrate the two items in the form of $(1-\alpha) \times item_1 + \alpha \times item_2$. It can be described specifically as follows:

$$r_{i,j}^{B}(t+1) = \frac{2}{3} \times \left[(1-\alpha) + \frac{\alpha}{\widehat{AD}_{i,j}^{B}}\right] \times r_i^{P}(t) \tag{1}$$

$$\widetilde{r}_{i,j,k}^{C}(t+1) = \frac{2}{3} \times \left[(1-\alpha) + \frac{\alpha}{\widehat{AD}_{i,j,k}^{C}}\right] \times r_{i,j}^{B}(t) \tag{2}$$

where $r_i^P(t)$, $r_{i,j}^B(t)$ and $\widetilde{r}_{i,j,k}^C(t+1)$ represent the range of PA_i, $BA_{i,j}$ and $CA_{i,j,k}$ in each iteration, respectively. $\widehat{AD}_{i,j}^B$ and $\widehat{AD}_{i,j,k}^C$ refer to the correction factor $\widehat{AD}$ of $BA_{i,j}$ and $CA_{i,j,k}$. The index i, j and k indicate that the current parameter attaches to PA_i, BA_j and CA_k.

When newborn CAs are distributed, the sampling information will also affect the property and range of PAs and the changing of PAs′ property will also affect the property and range of CAs′. This process is called as self-correcting in range which can be described as:

$$r_i^P(t+1) = (1-\alpha) \times r_i^P(t) + \alpha \times \frac{r_i^P(t)}{\widehat{AD}_i^P} \tag{3}$$

$$r_{i,j,k}^{C}(t+1) = \left[(1-\alpha) + \frac{\alpha}{\widehat{AD}_i^P}\right] \times \widetilde{r}_{i,j,k}^{C}(t+1) \tag{4}$$

where $\widetilde{r}_{i,j,k}^C(t+1)$ and $r_{i,j,k}^C(t+1)$ are the range of $CA_{i,j,k}$ before and after the self-correction. The equation set $(1) \sim (4)$ is also called the range-dividing principle in DDEA. Larger range caused by smaller $\widehat{AD}$ will make weak agents explore outward to discover more dominant position. On the contrary, smaller range caused by larger $\widehat{AD}$ will converge agents to exploit the emergent region fast.

The scale of newborn agents is also determined by two items. The first one is the scale which is the same as their parents, which means the child agents will inherit s_{parent} from their parents. The second one is the regulated scale with $\widehat{RD}$ or $\widehat{DC}$, which means the child agents will study from the sampling information and adjust their scale toward $\widehat{RD} \times r_{parent}$ or $\widehat{DC} \times r_{parent}$. Then, the scale of those agents of which the fitness growth is larger will be set to a larger value and vice versa. To make information fusion, α is also introduced as a study factor to integrate the two items in the form of $(1-\alpha) \times item_1 + \alpha \times item_2$. It can be described specifically as follows:

$$s_{i,j}^B(t+1) = \left[(1-\alpha) + \alpha \times \widehat{RD}_{i,j}^B\right] \times s_i^P(t) \tag{5}$$

$$\widetilde{s}_{i,j,k}^C(t+1) = \left[(1-\alpha) + \alpha \times \widehat{RD}_{i,j,k}^C\right] \times s_{i,j}^B(t) \tag{6}$$

where $s_i^P(t)$, $s_{i,j}^B(t)$ and $\widetilde{s}_{i,j,k}^C(t+1)$ represent the scale of PA_i, $BA_{i,j}$ and $CA_{i,j,k}$ in each iteration, respectively. $\widehat{RD}_{i,j}^B$ and $\widehat{RD}_{i,j,k}^C$ refer to the correction factor $\widehat{RD}$ of $BA_{i,j}$ and $CA_{i,j,k}$, respectively. The index i, j and k indicate that the current parameter attaches to PA_i, BA_j and CA_k, respectively.

When newborn CAs are distributed, the sampling information will also affect the property and the scale of PAs, and the changing of PAs′ property will affect the property and scale of CAs′. This process is called as self-correcting in scale which can be described as:

$$s_i^P(t+1) = \left[(1-\alpha) + \alpha \times \widehat{DC}_i^P\right] \times s_i^P(t) \tag{7}$$

$$s_{i,j,k}^C(t+1) = \left[(1-\alpha) + \alpha \times \widehat{DC}_i^P\right] \times \widetilde{s}_{i,j,k}^C(t+1) \tag{8}$$

where $\widetilde{s}_{i,j,k}^C(t+1)$ and $s_{i,j,k}^C(t+1)$ are the scale of $CA_{i,j,k}$ before and after the self-correction, respectively. The reason to select $\widehat{DC}_i^P$ as a self-correcting factor is that the focus of $\widehat{DC}$ will transit from $\widehat{RD}$ to $\widehat{AD}$ and large number of agents should be used to exploit the area where there may be the optimal agent in the later phase of optimization. While the algorithm runs, the focus of larger scale agents will translate from complex area to dominant area. Thus, DDEA can carry out specific exploration at the earlier stage of optimization and develop fast targeted exploitation at the later phase of optimization. In addition, a swarm scale floor (SSF) is introduced to reduce quickly the number of agents in the area where there is little growth in fitness. It not only saves sample resource but also maintains global exploration with fewer agents, which can be formulated as:

$$s(t+1) = \widehat{SSF}, \text{ if } \widehat{RD} \leq 0 \tag{9}$$

The equation set $(5) \sim (9)$ is also called the scale-setting principle in DDEA. Larger scale caused by larger $\widehat{RD}$ or $\widehat{DC}$ will not only reduce the loss of information in volatile region but also strengthen the ability of both exploration and exploitation towards potential advantage areas. On the contrary, smaller scale caused by smaller $\widehat{RD}$ or $\widehat{DC}$ will save much time that might be wasted in poor areas.

Quick-shift between exploration and exploitation is an important character of DDES. The degree of exploration and exploitation can be regulated according to the special condition in each partition. In addition, the population of agents in DDES is adaptively adjusted according to the need of exploration and exploitation. This mechanism not only makes DDES keep exploring globally but also gives DDES sufficiently fast convergence focusing on emergent areas. When a local optimal area is found by exploration in a partition, much more CAs will be generalized to exploit fast in the area. After the fast convergency conducted by CAs, a few of better CAs will be remained and transformed to the corresponding BAs as personal best agents. Other dispersive CAs will keep exploration in the corresponding partition. Thus, a preferable trade-off between exploration and exploitation will be acquired in each partition, and the personal best agents in each partition can reflect the distribution of local extremum to some extent.

1.2 Subregion dividing

Subregion is defined as an emergent area where the global optimum will be probably found. It is assigned in this paper as an area where agents can find much higher fitness, such as the inherent range around the current global best agent in DDES. The higher fitness in subregions indicates that the global optimum probably exists in these smaller regions. The difficulty in deciding such a subregion is that the existence of global optimum cannot be ensured, just as that the reliability of result obtained by swarm optimization algorithms cannot be ensured. So proper exploration should be considered around the subregions. Thus, the inherent range around the current global best agent of DDES appears to be too small to meet the requirement of exploration.

Selecting a flexible extent for a subregion seems to be advisable in improving the ability of exploration. In view of the optimum estimation of BAs and the inherent extensive exploration of DDES′s PA(s), personal best positions will be picked out in each partition and construct a smaller but better region of which the range is labeled as $\widehat{R}$. It can be determined through calculation as shown in (10):

$$\widehat{R} = (\max.X_{P\_best\_i}) - (\min.X_{P\_best\_i}) \tag{10}$$

where $X_{P\_best\_i}$ means the decision variable of a personal best result in partition i. 'max.' and 'min.' mean to find the maximum and minimum of each element in vector $X_{P\_best\_i}$ respectively. To establish a subregion, the current global agent whose position is labeled as $\widehat{P}$ should be focused on. So the subregion presented in this paper consists of $\widehat{R}$ and $\widehat{P}$. Taking $\widehat{P}$ as the center of a region and $\widehat{R}$ as the range of this region around point $\widehat{P}$, then a subregion will be got.

Subregion plays an important role in regulating exploration and improving exploitation, which will concentrate agents and accelerate optimizing speed especially when dealing with complex problems whose objective functions are simple. It is just like to plot out a key area on the whole feasible region, which will make optimization algorithm focus on a smaller area where there is probably a globally optimal result. Thus, the exploration in subregions with different sizes is flexible. The smaller a subregion is, the more targeted exploration is. If the subregion is small enough and contains the global optimum, then exploration in this subregion can guide exploitation through local extrema to the global optimum in suitable steps. Equation (10) also shows that the least value of $\widehat{R}$ will be limited to the division of partitions because the distribution of each personal best agent is limited to the partition to which the agent belongs. The focusing effect will be more obvious if the quantity of partitions is less in DDES, and vice versa.

2 The iteration

As discussed in Subsection 1.2, the focusing effect of a subregion found in global area by much more partitions in DDES will be not obvious. But commonly large numbers of partitions are required to carry on wide exploration against some more complex engineering problems. Thus, the focusing effect of subregion will not be made full use of if the subregion is just adopted in global area through the whole process of DDES. On the other hand, the complexity of objective function in most engineering problems is far from that of mixed constraints. So strong exploration in DDES should find a much smaller subregion which will play an important role in improving the efficiency of optimization algorithm. The issue is how to combine the advantage of both DDES and subregion to form an algorithm whose exploration and exploitation are both higher.

If subregions found by DDES in global area will not work in that case, alteration between calling DDES in the latest subregion and assigning a smaller subregion for next calling may serve the purpose. It starts by dividing the feasible region into partitions using PAs of DDES, then global best agent and personal best agents found by BAs and CAs will construct a subregion. Once a subregion has been found, the next round of dividing in the subregion will be carried on by DDES. It can be illustrated as Fig. 2 shows.

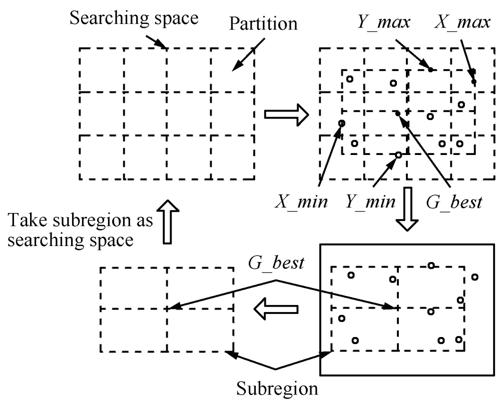

Fig. 2 The iteration caused by subregion

As DDES leads the iteration from a searching space to a subregion then to a new smaller searching space, we call this process as iterative DDES which can be described as follows:

1) To divide searching space into a number of partitions.

2) To find the personal best individual (P_best) of each partition by DDES.

3) To calculate $\widehat{R}$ and find the global best individual (G_best) according to these personal best agents.

4) To judge whether the breaking conditions are met. If it is, then jump to the last step, otherwise, continue downwards.

5) To plot out a subregion centered with G_best.

6) Take the subregion as a new searching space in which a new round of DDES will be carried on and then jump to the first step.

7) Output the results.

Obviously, a precondition is that the exploration of the optimization algorithm should meet the requirement of finding the character of extreme value distribution. Fortunately, DDES meets the requirement because of its strong exploration. So if DDES comes along with the redividing by subregion mentioned above, we will have an opportunity to create a successful optimization algorithm, iterative dynamic diversity evolutionary algorithm (IDDEA), for constrained optimization. The next task is to find a proper penalty function for IDDEA.

3 Infimum penalty and IDDEA process

Although penalty function method is one of the most common methods to solve constrained optimization problems, the design of penalty item should be fit for the algorithm by which we will deal with practical optimization problems. In order to avoid the complication caused by adding penalty term to the objective function, the penalty function will be departed away from the objective function and introduced into IDDEA as a classification condition and a fitness calculation.

The general constrained problem formulation that is also called the primal problem can be stated as follows:

$$\begin{aligned} \min\quad & f(\boldsymbol{x}) \\ \text{s.t.}\quad & g_i(\boldsymbol{x}) \le 0, && i = 1, \cdots, m \\ & h_i(\boldsymbol{x}) = 0, && i = m+1, \cdots, m+l \\ & \boldsymbol{x} = (x_1, x_2, \cdots, x_n) \in \mathbf{R}^n \end{aligned}$$

where $\boldsymbol{x}$ represents a vector of n real variables subject to a set of m inequality constraints $g(\boldsymbol{x})$ and a set of l equality constraints $h(\boldsymbol{x})$. The penalty function $p(\boldsymbol{x})$ proposed here can be defined as:

$$p(\boldsymbol{x}) = \sum_{i=1}^{m} \phi(g_i(\boldsymbol{x})) + \sum_{i=m+1}^{m+l} \varphi(h_i(\boldsymbol{x})) \tag{11}$$

where

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \le 0 \end{cases} \tag{12}$$

$$\varphi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x = 0 \\ -x, & \text{if } x < 0 \end{cases} \tag{13}$$

Penalty function $p(\boldsymbol{x})$ will be used in IDDEA to judge whether an agent is in the feasible region. Once an agent is updated by IDDEA, $p(\boldsymbol{x})$ will be called to check the decision variable represented by this agent. If the function result is zero, a feasible decision variable is got. Otherwise, the function result will replace the objective function value and be regarded as the fitness of the agent after subtracting from the minimum fitness of those agents in feasible regions. It can be expressed as what is shown in (14), the fitness function $\widehat{f}(\boldsymbol{x})$:

$$\widehat{f}(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}), & \text{if } p(\boldsymbol{x}) = 0 \\ \min(\widetilde{f}) - p(\boldsymbol{x}), & \text{if } p(\boldsymbol{x}) > 0 \end{cases} \tag{14}$$

where $\widetilde{f}$ represents the fitness of agents in feasible regions. As the minimum fitness of feasible agents is selected to construct the base of a penalty function when calculating the fitness of unfeasible agents, the penalty function is called an infimum penalty function.

The agents which include PAs, BAs and CAs in IDDEA are divided by penalty function $p(\boldsymbol{x})$ into two kinds: SAs that satisfy the constraint and UAs that unsatisfy the constraint. The fitness of SAs is got from objective function $f(\boldsymbol{x})$ in the first part of fitness function $\widehat{f}(\boldsymbol{x})$, while the fitness of UAs is calculated according to the second part of fitness function $\widehat{f}(\boldsymbol{x})$. As shown in (14), the fitness of UAs consists of 'state penalty' and 'constraint penalty'. State penalty refers to the minimum fitness (or objective function) found by SAs in the current searching results. It indicates a dynamic penalty boundary basing on the actual situation. So state penalty can avoid too little or too much punishment. Constraint penalty refers to the UAs′ distance from the constraint boundary. The further one agent is out of the feasible region, the more fitness penalty it will suffer. So UAs will evolve towards the feasible region under such selection pressure.

By introducing the infimum penalty function into the IDDEA proposed in Section 2, we will get an efficient search algorithm against constrained optimization. Then, the execution flow of IDDEA can be illustrated in Fig. 3

4 Benchmark problems

The performance of the proposed IDDEA was investigated on problems frequently employed in the published literature, where other state-of-the-art methods developed for constrained optimization are compared. As IDDEA was used to find the maximum value, the opposite value of objective function was selected as objective value in the following evaluation process. The algorithms had been implemented in MATLAB 7.10. Four test problems proposed by different authors have been chosen to illustrate the applicability and the efficiency of the implemented algorithm. The details of initialization conditions are described in the following subsections.

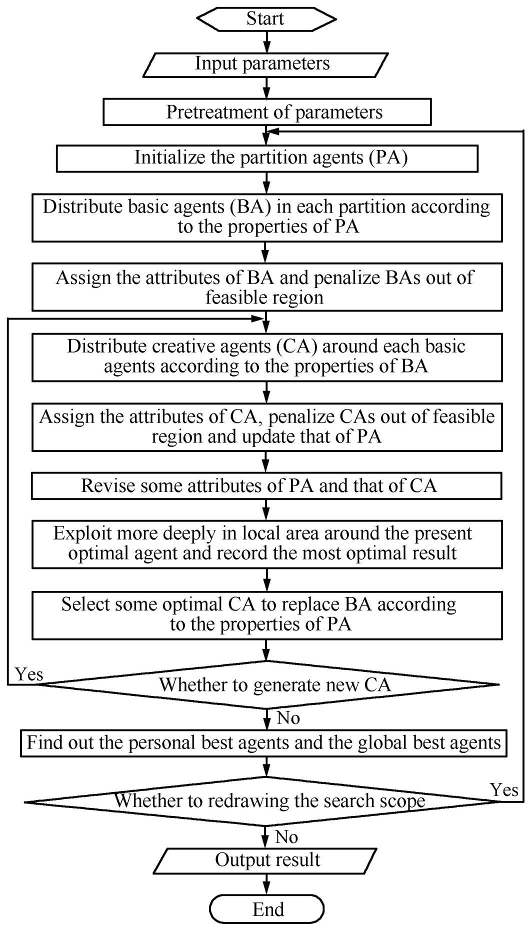
Start
Input parameters
Pretreatment of parameters
Initialize the partition agents (PA)
Distribute basic agents (BA) in each partition according to the properties of PA
Assign the attributes of BA and penalize BAs out of feasible region
Distribute creative agents (CA) around each basic agents according to the properties of BA
Assign the attributes of CA, penalize CAs out of feasible region and update that of PA
Revise some attributes of PA and that of CA
Exploit more deeply in local area around the present optimal agent and record the most optimal result
Select some optimal CA to replace BA according to the properties of PA
Yes
Whether to generate new CA
No
Find out the personal best agents and the global best agents
Whether to redrawing the search scope
Yes
No
Output result
End

Fig. 3 The process of IDDEA

4.1 Himmelblau′s function

Himmelblau′s function originally proposed by Himmelblau[36] is a common benchmark problem in nonlinear constrained optimization. This problem consists of five design variables, six nonlinear constraints, and ten boundary conditions as follows:

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + \\ & \qquad 37.293239x_1 - 40792.141 \\ \text{s.t.} \quad & 0 \le g_1(\boldsymbol{x}) \le 92 \\ & 90 \le g_2(\boldsymbol{x}) \le 110 \\ & 20 \le g_3(\boldsymbol{x}) \le 25 \end{aligned}$$

where

$$\begin{aligned} g_1(\boldsymbol{x}) = {} & 85.334407 + 0.0056858x_2x_5 + \\ & 0.0006262x_1x_4 - 0.0022053x_3x_5 \\ g_2(\boldsymbol{x}) = {} & 80.51249 + 0.0071317x_2x_5 + \\ & 0.0029955x_1x_2 + 0.0021813x_3^2 \\ g_3(\boldsymbol{x}) = {} & 9.300961 + 0.0047026x_3x_5 + \\ & 0.0012547x_1x_3 + 0.0019085x_3x_4 \end{aligned}$$

and

$$78 \leq x_1 \leq 102, \quad 33 \leq x_2 \leq 45, \quad 27 \leq x_3 \leq 45$$
$$27 \leq x_4 \leq 45, \quad 27 \leq x_5 \leq 45$$

Several attempts have been made to solve this problem. Take the list given by Nema et al.[3] as an instance, Himmelblau used a generalized reduced gradient method to search for the optimal solution and acquired a result of −30 373.94. This problem was also investigated by Runarsson and Yao[37], where an evolutionary strategy with stochastic ranking was suggested. They reported a better result of −30 665.53. Gen and Cheng[38] have tackled this problem by using an improved GA and the best solution they obtained was −30 183.57. Recently, Nema et al.[3] have published a more excellent result of −30 665.57 using HCP algorithm.

In our approach, the IDDEA found an optimal objective function value of −30 665.54 using 60 times of contraction led by subregions. The learning factor and the initial agents size were chosen as 0.5 and 3^D, respectively (D refers to the dimension of decision variable). The optimal results of Himmelblau′s problem found by HCP and IDDEA were compared in Table 1. The excellent result, −30 665.53867, found by IDDEA was a mean fitness value. All the fitness values for 10 independent runs of IDDEA are shown in Table 2. The results stabilize around the mean fitness except for slight fluctuations caused by the limitation of searching precision. If we increase the times of contraction, the searching precision of this algorithm will be improved because of much smaller subregions. Even though the number of iterations was set to a value that was not too high, the result of IDDEA was satisfactory.

4.2 Minimization of the weight of a tension/compression string

This engineering design problem taken from Belegundu[39] is much more complicated. It describes the minimization of the weight of the tension/compression spring subject to constraints on minimum deflection, shear stress, surge frequency, and limits on outside diameter. The mathematical formulation of the problem is stated as follows:

$$\begin{aligned}
\min \quad & f(\boldsymbol{x}) = (x_3+2)x_2x_1^2 \\
\text{s.t.} \quad & g_1(\boldsymbol{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \leq 0 \\
& g_2(\boldsymbol{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0 \\
& g_3(\boldsymbol{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0 \\
& g_4(\boldsymbol{x}) = \frac{x_1+x_2}{1.5} - 1 \leq 0
\end{aligned}$$

Many researchers have previously investigated this minimization problem. Belegundu[39] proposed an approach based on Lagrange multipliers technique to deal with this challenging problem, while Arora[40] attempted a numerical optimization method called constraint correction at constant cost. In addition, Coello and Montes[41] obtained better known results by using a dominance-based selection scheme to incorporate constraints into the fitness function of a genetic algorithm. More wonderful results were found by Nema et al.[3] basing on a hybrid cooperative search algorithm for constrained optimization. Their results are shown in Table 3 comparing with the result obtained by the proposed algorithm in this paper, when using 40 times of contraction led by subregions with learning factor as 0.5 and initial agents size as 3^D (D refers to the dimension of decision variable).

Table 1 The comparison results of Himmelblau′s problem between HCP and IDDEA

Optimization algorithm	HCP[3]	IDDEA
Optimal objective $f(\boldsymbol{x})$	−30 665.57	−30 665.53867
Decision variables $\boldsymbol{x}$	[78.0027, 32.9992, 29.9809, 45.0000, 36.8109]	[78.0000, 33.0000, 29.9953, 45.0000, 36.7758]
Constraint $0 \leq g_1(\boldsymbol{x}) \leq 92$	92.0053 (No)	92.0000 (Yes)
Constraint $90 \leq g_2(\boldsymbol{x}) \leq 110$	98.8467 (Yes)	98.8405 (Yes)
Constraint $20 \leq g_3(\boldsymbol{x}) \leq 25$	19.9999 (No)	20.0000 (Yes)
Constraint $78 \leq x_1 \leq 102$	Yes	Yes
Constraint $33 \leq x_2 \leq 45$	No	Yes
Constraint $27 \leq x_3 \leq 45$	Yes	Yes
Constraint $27 \leq x_4 \leq 45$	Yes	Yes
Constraint $27 \leq x_5 \leq 45$	Yes	Yes

Table 2 The optimal results found by IDDEA across Himmelblau′s problem

No.	Min $f(\boldsymbol{x})$	No.	Min $f(\boldsymbol{x})$
1	−30 665.53867178001	6	−30 665.53867178327
2	−30 665.53867178031	7	−30 665.53867178321
3	−30 665.53867154942	8	−30 665.53867177717
4	−30 665.53867178278	9	−30 665.53867178329
5	−30 665.53867178299	10	−30 665.53867178329

4.3 Pressure vessel design

This optimization problem put forward by Sandgren[42] aims to minimize the cost of materials for forming and welding of a pressure vessel. There are four design variables: T_s (Thickness of the shell), T_h (Thickness of the head), R (Inner radius), and L (Length of the cylindrical section of the vessel, not including the head). T_s and T_h are integer multiples of 0.0625 in., which are the available thicknesses of rolled steel plates, and R and L are continuous. This optimization problem can be described as:

$$\begin{aligned}\min\quad & f(\boldsymbol{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + \\ & \quad 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\ \text{s.t.}\quad & g_1(\boldsymbol{x}) = 0.0193x_3 - x_1 \le 0 \\ & g_2(\boldsymbol{x}) = 0.00954x_3 - x_2 \le 0 \\ & g_3(\boldsymbol{x}) = 1\,296\,000 - \pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 \le 0 \\ & g_4(\boldsymbol{x}) = x_4 - 240 \le 0\end{aligned}$$

and

$$0.0625 \le x_1 \le 6.1875,\qquad 0.0625 \le x_2 \le 6.1875$$
$$10 \le x_3 \le 200,\qquad 10 \le x_4 \le 200$$

Previous researchers took this design problem as a mixed integer problem. An evolutionary algorithm named as GeneAS was used by Deb[43] to solve the problem. Coello and Montes[44] also attempted to solve this problem by using GA with dominance-based tournament selection to handle the constraints. An improved PSO was proposed by He et al.[4] to deal with pressure vessel design. A better result was obtained by Nema et al.[3] by using HCP. The results found by these different algorithms are shown in Table 4 comparing with a wonderful result obtained by the present algorithm. When using 40 times of contraction leaded by subregions with learning factor as 0.5 and initial agents size as 3^D (D refers to the dimension of decision variable), more wonderful design variables are found by IDDEA as shown below:

$$x_1 = 0.7781686497708,\qquad x_2 = 0.3846491690908$$
$$x_3 = 40.3196190969763,\qquad x_4 = 199.9999948102470$$

with optimal result 5 885.33.

4.4 Welded beam design

The optimization problem introduced by Ragsdell and Phillips[45] aims to find the minimum cost design of a structural welded beam. The problem consists of seven linear and nonlinear constraints on (g_1) shear stress (τ), (g_2) bending stress in the beam (σ), (g_3, g_4, g_5) side constraints, (g_6) end deflection of the beam (δ), and (g_7) buckling load on the bar (P_c). The problem with four design variables is described as follows:

$$\begin{aligned}\min\quad & f(\boldsymbol{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \\ \text{s.t.}\quad & g_1(\boldsymbol{x}) = \tau(\boldsymbol{x}) - \tau_{\max} \le 0 \\ & g_2(\boldsymbol{x}) = \sigma(\boldsymbol{x}) - \sigma_{\max} \le 0 \\ & g_3(\boldsymbol{x}) = x_1 - x_4 \le 0 \\ & g_4(\boldsymbol{x}) = 0.10471x_1^2 + \\ & \quad 0.04811x_3x_4(14.0 + x_2) - 5.0 \le 0 \\ & g_5(\boldsymbol{x}) = 0.125 - x_1 \le 0 \\ & g_6(\boldsymbol{x}) = \delta(\boldsymbol{x}) - \delta_{\max} \le 0 \\ & g_7(\boldsymbol{x}) = P - P_c(\boldsymbol{x}) \le 0\end{aligned}$$

and

$$0.1 \le x_1 \le 2.0,\qquad 0.1 \le x_2 \le 10.0$$
$$0.1 \le x_3 \le 10.0,\qquad 0.1 \le x_4 \le 2.0$$

where

$$\tau(\boldsymbol{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$
$$\tau' = \frac{P}{\sqrt{2}x_1x_2},\quad \tau'' = \frac{MR}{J},\quad M = P\left(L + \frac{x_2}{2}\right)$$
$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$
$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$
$$\sigma(\boldsymbol{x}) = \frac{6PL}{x_4x_3^2},\quad \delta(\boldsymbol{x}) = \frac{4PL^3}{Ex_3^3x_4}$$
$$P_c(\boldsymbol{x}) = \frac{4.013\sqrt{EG(x_3^2x_4^6/36)}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

Table 3 Comparison of the results for the minimization of cost of the weight of a tension spring

Algorithm	IDDEA	HCP[3]	Coelle and Montes[41]	Arora[40]	Belegundu[39]
x_1	0.05160011	0.051987	0.051989	0.053396	0.050000
x_2	0.35458147	0.363964	0.363965	0.399180	0.315900
x_3	11.41532664	10.890521	10.890522	9.185400	14.25000
g_1	−5.5E-007	−0.0014	−0.0000	0.0000	−0.0000
g_2	−2.5E-007	0.0000	−0.0000	−0.0000	−0.0037
g_3	−4.04954161	−4.0611	−4.0613	−4.1238	−3.9383
g_4	−0.72921228	−0.7226	−0.7226	−0.6982	−0.7560
$f(\boldsymbol{x})$	0.01266539	0.012679	0.012681	0.012730	0.012833

Table 4 Comparison of the results for the minimization of cost in pressure vessel design

Algorithm	IDDEA	HCP[3]	He et al.[4]	Coelle and Montes[44]	Deb[43]
$f(\boldsymbol{x})$	5 885.33	6 059.69	6 059.71	6 059.94	6 410.38

Table 5 Comparison of the results for the minimization of cost in welded beam design

Algorithm	IDDEA	HCP[3]	He et al.[4]	Ray and Liew[46]	Ragsdell and Phillips[45]
$f(\boldsymbol{x})$	1.8616	2.3809	2.3809	2.3854	2.3859

The numerical parameters for this model are chosen as:

$$P = 6000\,\text{lb.}, \quad L = 14\,\text{in.}$$
$$E = 30 \times 10^6\,\text{psi}, \quad G = 12 \times 10^6\,\text{psi}$$
$$\tau_{\max} = 13600\,\text{psi}, \quad \sigma_{\max} = 30000\,\text{psi}$$
$$\delta_{\max} = 0.25\,\text{in.}$$

There has been much discussion on this problem by many researchers with various methods. Ragsdell and Phillips[45] handled the problem with geometric programming. It was also dealt with by Ray and Liew[46] using a society and civilization algorithm. He et al.[4] had tackled this optimization problem by using an improved PSO. A well known result was reported by Nema et al.[3] with their HCP algorithm. The results found by them are shown in Table 5 comparing with the result searched by the present algorithm. The design variables found by IDDEA with the same initial conditions used in Section 4.3 are shown as following:

$$x_1 = 0.244368999403763, \quad x_2 = 3.040294849243054$$
$$x_3 = 8.291470822579198, \quad x_4 = 0.244369009286497$$

with optimal result 1.8616.

5 Comparison and analysis

In order to further assess the benefits from the proposed algorithm, the four problems referred by Nema et al.[3] are considered here for the purpose of comparing with other evolutionary algorithms published in the literature such as HCP and the common PSO. When the iteration number is 200, learning factor is 0.5 and initial agents size is 3^D (D refers to the dimension of decision variable) for 100 executions, the proportion of IDDEA converged to global optimum is over 87 % across these four problems. But the common PSO has a lower proprotion of 71 ∼ 83% when the maximum iteration number is 2 000 with swarm size 40. The proportion of HCP put forward by Nema et al.[3] is 81 % in the search process for Himmelblau problem. When applying the proposed method to these four problems, the convergence rate is found to be better than HCP and similar to the improved PSO as shown from Fig. 4 ∼ Fig. 7. Although the rate of convergence of IDDEA is sometimes lower than PSO, the final optimization results are mostly the best as shown in Table 6. Thus, the proposed IDDEA in this paper performs well in both optimization accuracy and rapidity.

This paper presents a method of swarm evolution with multi-agents under the three layers proxy mode of DDES and a novel optimum estimation strategy to carry out subregion converging iteration. Simulation results based on well-known constrained engineering design problems suggest that the IDDEA is capable of locating the global optimum with better convergence rate. It is because the proposed algorithm has strong exploration ability by adopting the TLPM of DDES, which reduces the possibility of being trapped in local optimum. The novel optimum estimation strategy with $\widehat{AD}$, $\widehat{RD}$ and $\widehat{DC}$ gives a better guide for multi-agents toward dominant area, which is coupled with the novel evolution model of agents′ range and scale and provides sufficient basis for the subregion dividing. Thus, IDDEA is capable of finding a smaller but more dominant area under the framework of subregion dividing and infimum penalty, and converging to the global optimum quickly. It is worth noting that IDDEA is more fit for solving problems with complex constrains around the global optimum in a much smaller area.

The proposed algorithm is aimed at avoiding being trapped in local optimum which is formed by the complex constraint condition nearing the global optimum. Other kinds of constrained optimization problems such as the problems with high dimensional design variable and equality constraint should be considered in the future. Some engineering design problems solved in this paper have revealed that IDDEA is fit for the problems with lower dimension and inequality constraints. The Himmelblau problem, one of CEC 06 (problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization), is used here for testing. Other constrained optimization problems of CEC 06 which have lower dimension and inequality constraints are also tested by IDDEA comparing with PSO as shown in Table 7.

6 Conclusion and future work

As there are many engineering problems whose objective functions are simple while the constraint conditions are complex, the personal best agents in each iteration of IDDEA can confirm a subregion where the global optimization could be obtained. Once a subregion is found, another

Table 6 Comparison of the results for the minimization in four problems

Problems	IDDEA	HCP[3]	He et al.[4]	PSO
Himmelblau	−30 665.54	−30 665.57	–	−30 665.54
Tension spring	0.012665	0.012679	–	0.012856
Pressure vessel	5 885.33	6 059.69	6 059.71	6 038.55
Welded beam	1.8616	2.3809	2.3809	1.8698

Table 7 Testing across constrained problems of CEC 06

Prob.	n	$f(\boldsymbol{x}^*)$	IDDEA	PSO
g04	5	−30 665.5386717834	−30 665.5386717832	−30 665.5386717828
g06	2	−6 961.8138755802	−6 961.8138755799	−6 961.8138755801
g08	2	−0.0958250415	−0.0958250414	−0.0958250414
g12	3	−1.0000000000	−1.0000000000	−1.0000000000
g24	2	−5.5080132716	−5.5080132716	−5.5080132716

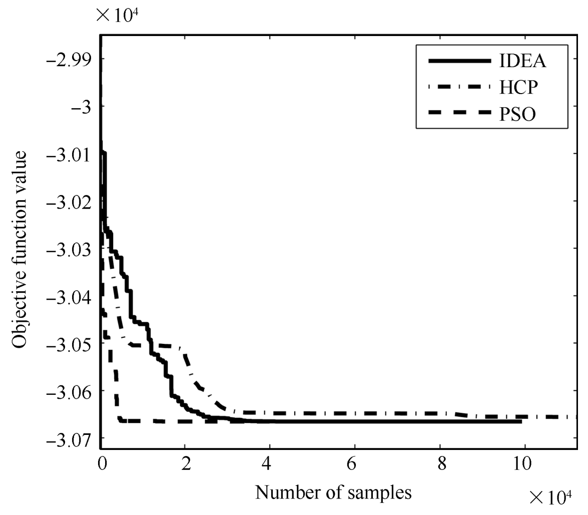

Fig. 4 Evolution plots for Himmelblau problem

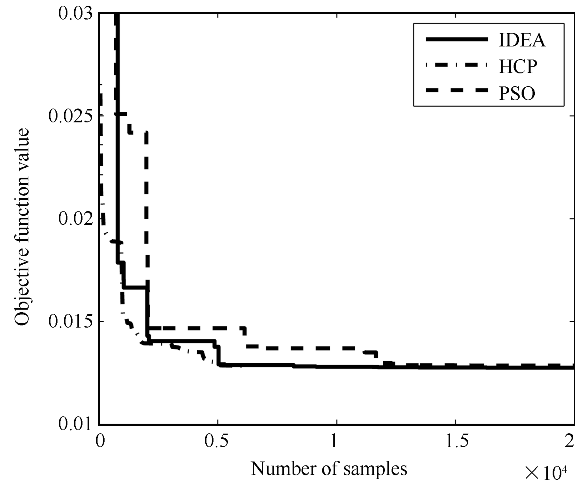

Fig. 5 Evolution plots for spring′s tension problem

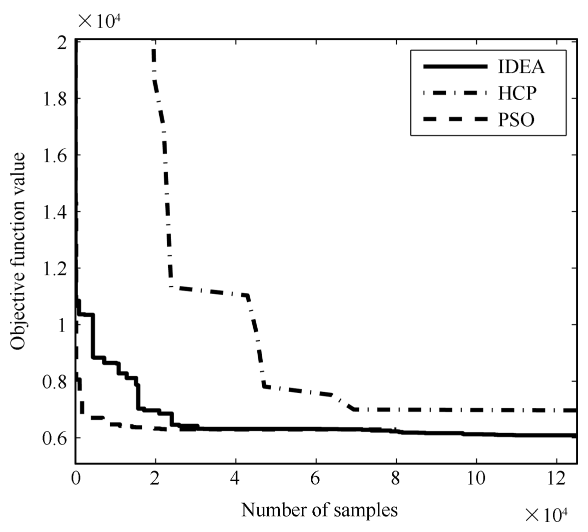

Fig. 6 Evolution plots for pressure vessel design

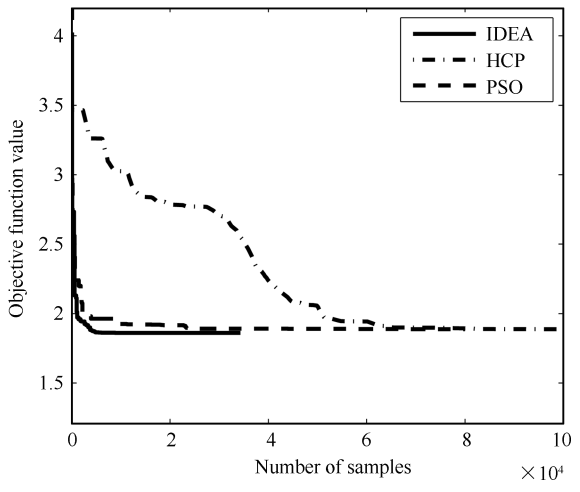

Fig. 7 Evolution plots for welded beam design

specific exploration and exploitation with faster and more accurate convergence will be carried out in the subregion. When the subregion converges to a much smaller area, the global optimum will be hard to be lost. Furthermore, the infimum penalty function is not added to the objective function directly but constitutes the IDDEA as a correction term. It selects the worst agent as a reference term and adds an infimum penalty term into IDDEA, which can avoid the difficulty in designing those parameters or weights in other penalty based methods.

The performance of IDDEA proposed in this paper was evaluated by comparing its accuracy with other well known optimization algorithms. The more optimal results presented in Section 4 confirmed the strong global searching ability of IDDEA in dealing with complex constrained optimization problems. It also revealed that the dynamic subregions and the infimum penalty term not only kept the capability of global search but also enhanced the convergence which was correlative with the precision of optimization results. Thus, the improvements presented in this article on original DDES can also be used in other simplex evolutionary algorithms to extend their application scope to complex constrained optimization successfully.

To make IDDEA adapt to more comprehensive environment easily is an important goal in future study. The generalization capability of IDDEA needs to be improved further, which is difficult and not practical sometimes. On the other hand, parameter matching for IDDEA should be systematized comprehensively according to more types of problems. Although the engineering design problems and the benchmark constrained optimization problems have been resolved well, many more experiments and practices are needed for identifying the most suitable range for the application of IDDEA. To improve the present algorithm, other state-of-the-art methods dealing with constraints can be used to combine with the infimum penalty function. As IDDEA is a simplex evolutionary algorithm, it can also be used in co-evolutionary algorithms to extend its application field.

References

1 Deb K. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 2000, **186**(2−4): 311−338

2 Farmani R, Wright J A. Self-adaptive fitness formulation for constrained optimization. *IEEE Transactions on Evolutionary Computation*, 2003, **7**(5): 445−455

3 Nema S, Goulermas J Y, Sparrow G, Helman P. A hybrid cooperative search algorithm for constrained optimization. *Structural and Multidisciplinary Optimization*, 2011, **43**(1): 107−119

4 He S, Prempain E, Wu Q H. An improved particle swarm optimizer for mechanical design optimization problems. *Engineering Optimization*, 2004, **36**(5): 585−605

5 Yang B, Chen Y P, Zhao Z L, Han Q Y. A master-slave particle swarm optimization algorithm for solving constrained optimization problems. In: Proceedings of the 6th World Congress on Intelligent Control and Automation. Dalian, China: IEEE, 2006. 3208−3212

6 Huang Xiao-Ling, Chai Tian-You. Particle swarm optimization for raw material purchasing plan in large scale ore dressing plant. *Acta Automatica Sinica*, 2009, **35**(5): 632−636 (in Chinese)

7 Wang Chun-Sheng, Wu Min, Cao Wei-Hua, He Yong. Intelligent integrated modeling and synthetic optimization for blending process in lead-zinc sintering. *Acta Automatica Sinica*, 2009, **35**(5): 605−612 (in Chinese)

8 Liu Gang, Lao Song-Yang, Yuan Can, Hou Lv-Lin, Tan Dong-Feng. OACRR-PSO algorithm for anti-ship missile path planning. *Acta Automatica Sinica*, 2012, **38**(9): 1528−1537 (in Chinese)

9 Wu Q, Law R, Wu E, Lin J X. A hybrid-forecasting model reducing Gaussian noise based on the Gaussian support vector regression machine and chaotic particle swarm optimization. *Information Sciences*, 2013, **238**: 96−110

10 Wu Q, Law R. Complex system fault diagnosis based on a fuzzy robust wavelet support vector classifier and an adaptive gaussian particle swarm optimization. *Information Sciences*, 2010, **180**(23): 4514−4528

11 Vural R A, Yildirim T, Kadioglu T, Basargan A. Performance evaluation of evolutionary algorithms for optimal filter design. *IEEE Transactions on Evolutionary Computation*, 2012, **16**(1): 135−147

12 Li X D, Yao X. Cooperatively coevolving particle swarms for large scale optimization. *IEEE Transactions on Evolutionary Computation*, 2012, **16**(2): 210−224

13 Blackwell T. A study of collapse in bare bones particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 2012, **16**(3): 354−372

14 Pehlivanoglu Y V. A new particle swarm optimization method enhanced with a periodic mutation strategy and neural networks. *IEEE Transactions on Evolutionary Computation*, 2013, **17**(3): 436−452

15 Chen W N, Zhang J, Lin Y, Chen N, Zhan Z H, Chung H, Li Y, Shi Y H. Particle swarm optimization with an aging leader and challengers. *IEEE Transactions on Evolutionary Computation*, 2013, **17**(2): 241−258

16 Naznin F, Sarker R, Essam D. Progressive alignment method using genetic algorithm for multiple sequence alignment. *IEEE Transactions on Evolutionary Computation*, 2012, **16**(5): 615−631

17 Pan Feng, Zhou Qian, Li Wei-Xing, Gao Qi. Analysis of standard particle swarm optimization algorithm based on Markov chain. *Acta Automatica Sinica*, 2013, **39**(4): 381−389 (in Chinese)

18 Liu Jian-Hua, Liu Guo-Mai, Yang Rong-Hua, Hu Wen-Yu. Analysis of interactivity and randomness in particle swarm optimization. *Acta Automatica Sinica*, 2012, **38**(9): 1471−1484 (in Chinese)

19 Wang Y, Cai Z X. Combining multiobjective optimization with differential evolution to solve constrained optimization problems. *IEEE Transactions on Evolutionary Computation*, 2012, **16**(1): 117−134

20 Wang Y, Cai Z X, Zhou Y R, Zeng W. An adaptive tradeoff model for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 2008, **12**(1): 80−92

21 Cai Z X, Wang Y. A multiobjective optimization-based evolutionary algorithm for constrained optimization. *IEEE Transactions on Evolutionary Computation*, 2006, **10**(6): 658−675

22 Krohling R A, dos Santos-Coelho L. Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2006, **36**(6): 1407−1416

23 Tessema B, Yen G G. An adaptive penalty formulation for constrained evolutionary optimization. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 2009, **39**(3): 565−578

24 Daneshyari M, Yen G G. Constrained multiple-swarm particle swarm optimization within a cultural framework. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 2012, **42**(2): 475−490

25 Venkatraman S, Yen G G. A generic framework for constrained optimization using genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 2005, **9**(4): 424−435

26 Wang Y, Jiao Y C, Li H. An evolutionary algorithm for solving nonlinear bilevel programming based on a new constraint-handling scheme. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2005, **35**(2): 221−232

27 Grefenstette J J. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 1986, **16**(1): 122−128

28 Whitley D, Starkweather T, Bogart C. Genetic algorithms and neural networks: optimizing connections and connectivity. *Parallel Computing*, 1990, **14**(3): 347−361

29 Weile D S, Michielssen E. Genetic algorithm optimization applied to electromagnetics: a review. *IEEE Transactions on Antennas and Propagation*, 1997, **45**(3): 343−353

30 Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks. New York, USA: IEEE, 1995. 1942−1948

31 Eberhart R, Kennedy J. A new optimizer using particle swarm theory. In: Proceedings of the 6th International Symposium on Micro Machine and Human Science. Nagoya, USA: IEEE, 1995. 39−43

32 Settles M, Soule T. Breeding swarms: a GA/PSO hybrid. In: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation. Washington, DC: ACM, 2005. 161−168

33 Robinson J, Sinton S, Rahmat-Samii Y. Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna. In: Proceedings of Antennas and Propagation Society International Symposium, 2002. New York, USA: IEEE, 2002. 314−317

34 Rahmat-Samii Y. Genetic algorithm (GA) and particle swarm optimization (PSO) in engineering electromagnetics. In: Proceedings of the 17th International Conference on Applied Electromagnetics and Communications, 2003. Dubrovnik, Croatia: IEEE, 2003. 1−5

35 Gao Wei-Shang, Shao Cheng, Gao Qin. Pseudo-collision in swarm optimization algorithm and solution: rain forest algorithm. *Acta Physica Sinica*, 2013, **62**(19): 20−35 (in Chinese)

36 Himmelblau D M, Clark B J, Eichberg M. *Applied Nonlinear Programming*. New York: McGraw-Hill, 1972.

37 Runarsson T P, Yao X. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 2000, **4**(3): 284−294

38 Gen M, Cheng R. *Genetic Algorithms and Engineering design*. New York: John Wily and Sons, 1997.

39 Belegundu A D. A Study of Mathematical Programming Methods for Structural Optimization [Ph.D. dissertation], University of Iowa, Iowa, 1982.

40 Arora J S. *Introduction to optimum design*. New York: McGraw-Hill, 1989.

41 Coello C A C, Montes E M. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, 2002, **16**(3): 193−203

42 Sandgren E. Nonlinear integer and discrete programming in mechanical design optimization. *Journal of Mechanical Design*, 1990, **112**(2): 223-229

43 Deb K. Geneas: a robust optimal design technique for mechanical component design. *Evolutionary Algorithms in Engineering Applications*. New York: Springer 1997. 497−514

44 Coello C A C, Montes E M. Use of dominance-based tournament selection to handle constraints in genetic algorithms. *Intelligent Engineering Systems through Artificial Neural Networks (ANNIE2001)*, 2001, **11**: 177−182

45 Ragsdell K M, Phillips D T. Optimal design of a class of welded structures using geometric programming. *Journal of Manufacturing Science and Engineering*, 1976, **98**(3): 1021−1025

46 Ray T, Liew K M. Society and civilization: an optimization algorithm based on the simulation of social behavior. *IEEE Transactions on Evolutionary Computation*, 2003, **7**(4): 386−396

GAO Wei-Shang Ph. D. candidate at the School of Control Science and Engineering, Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology (DUT). He is a member of the Institute of Advanced Control Technology in DUT, China. He received his B. E. degree in automation from North China University of Water Resources and Electric Power, China in 2006, and M. E. degree in electronic information from DUT, China, in 2009. His research interest covers evolutionary computation, swarm intelligence optimization, and intelligent control. Corresponding author of this paper.
E-mail: gao.weishang@hotmail.com

SHAO Cheng Professor in DUT. His research interest covers modeling and control of complex industrial process, integrated optimization control of continuous industrial process, robust adaptive control of nonlinear system, multiobjective optimization control technology and intelligent control theory and method.
E-mail: cshao@dlut.edu.cn