Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2013, Article ID 912056, 9 pages
http://dx.doi.org/10.1155/2013/912056

Research Article

A Cooperative Coevolutionary Cuckoo Search Algorithm for Optimization Problem

Hongqing Zheng[1] and Yongquan Zhou[1,2]

[1] *Guangxi Key Laboratory of Hybrid Computation and Integrated Circuit Design Analysis, Nanning, Guangxi 530006, China*
[2] *College of Information Science and Engineering, Guangxi University for Nationalities, Nanning, Guangxi 530006, China*

Correspondence should be addressed to Yongquan Zhou; yongquanzhou@126.com

Received 24 May 2013; Accepted 8 July 2013

Academic Editor: Xin-She Yang

Taking inspiration from an organizational evolutionary algorithm for numerical optimization, this paper designs a kind of dynamic population and combining evolutionary operators to form a novel algorithm, a cooperative coevolutionary cuckoo search algorithm (CCCS), for solving both unconstrained, constrained optimization and engineering problems. A population of this algorithm consists of organizations, and an organization consists of dynamic individuals. In experiments, fifteen unconstrained functions, eleven constrained functions, and two engineering design problems are used to validate the performance of CCCS, and thorough comparisons are made between the CCCS and the existing approaches. The results show that the CCCS obtains good performance in the solution quality. Moreover, for the constrained problems, the good performance is obtained by only incorporating a simple constraint handling technique into the CCCS. The results show that the CCCS is quite robust and easy to use.

1. Introduction

High dimension numerical optimization problems tend to be complex, and general basic intelligent algorithms are difficult to obtain the global optimal solution. In order to solve this problem, many improved methods are put forward, such as evolutionary programming made faster [1], orthogonal genetic algorithm [2], and good point set based genetic algorithm [3], and these methods have achieved good effect.

Recently, a novel heuristic search algorithm, called Cuckoo Search (CS) in [4], has been proposed by Yang and Deb in 2009. The CS is a search algorithm based on the interesting breeding behavior such as brood parasitism of certain species of cuckoos. Each nest within the swarm is represented by a vector in multidimensional search space; the CS algorithm also determines how to update the position of cuckoo laid egg. Each cuckoo updates it position of lay egg based on current step size via Lévy flights. It has been shown that this simple model has been applied successfully to continuous nonlinear function, engineering optimization problem in [5], and so forth. Intelligent algorithm is based on the selection of the fittest in biological systems which have evolved by natural selection over millions of years, between organisms in nature not only competition but also cooperation. Potter et al. earlier proposed a cooperative coevolutionary genetic algorithm to function optimization (CCGA) [6], and Bergh et al. apply this idea to the standard particle group Algorithm to construct a new collaborative model (CPSO-SK) [7]. Liu et al. proposed an organizational evolutionary algorithm for numerical optimization [8], Mu et al. put forward M-elite coevolutionary algorithm for numerical optimization [9], and Fister et al. proposed memetic artificial bee colony algorithm for large-scale global optimization [10]. They have obtained the good effect in the numerical optimization problem. Based on this idea and combining evolutionary operators, this paper proposes a new algorithm of solving high-dimensional unconstrained, constrained, and engineering optimization problem, namely, a cooperative coevolutionary cuckoo search algorithm (CCCS) algorithm. Population of the algorithm is divided into M groups, each group has a leader, by annexation and collaborative operation between different organizations, and uses the cuboids crossover operator, discrete crossover operator, flip crossover operator, and mutation operator to achieve the exchange of information between individuals, to promote the evolution of the population. Simulation experiments show that CCCS optimization ability is very strong, can well solve the unconstrained optimization, constrained optimization, and engineering optimization problems and so on.

The remainder of this paper is organized as follows: Section 2 briefly introduces the original cuckoo search algorithm. This is followed in Section 3 by new cooperative coevolutionary implements of the CS algorithm. Section 4 describes the definition of a constrained optimization and a unconstrained optimization problem in the penalty function approach. The results can be found and discussed in Section 5. Finally, some directions for future research are discussed in Section 6.

2. Original CS

CS is a heuristic search algorithm which has been proposed recently by Yang and Deb. The algorithm is inspired by the reproduction strategy of cuckoos. At the most basic level, cuckoos lay their eggs in the nests of other host birds, which may be of different species. The host bird may discover that the eggs are not it's own and either destroy the egg or abandon the nest all together. This has resulted in the evolution of cuckoo eggs which mimic the eggs of local host birds. To apply this as an optimization tool, Yang and Deb [4] used three ideal rules.

(1) Each cuckoo lays one egg, which represents a set of solution coordinates, at a time, and dumps it in a random nest.

(2) A fraction of the nests containing the best eggs, or solutions, will carry over to the next generation.

(3) The number of nests is fixed, and there is a probability that a host can discover an alien egg. If this happens, the host can either discard the egg or the nest, and this results in building a new nest in a new location.

This algorithm uses a balanced combination of a local random walk and the global explorative random walk, controlled by a switching parameter p_a. The local random walk can be written as

$$x_i^{t+1} = x_i^t + \partial s \otimes H(p_a - \varepsilon) \otimes (x_j^t - x_k^t), \qquad (1)$$

where x_j^t and x_k^t are two different solutions selected randomly by random permutation, $H(u)$ is a Heaviside function, ε is a random number drawn from a uniform distribution, and s is the step size. On the other hand, the global random walk is carried out by using Lévy flights

$$x_i^{t+1} = x_i^t + \partial L(s, \lambda), \qquad (2)$$

where

$$L(s, \lambda) = \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}}, \qquad (s \gg s_0 > 0). \qquad (3)$$

Here $\partial > 0$ is the step-size-scaling factor, which should be related to the scales of the problem of interests. In most cases, we can use $\partial = O(L/10)$, and $\partial = O(L/100)$ can be more effective and can avoid flying too far.

3. A Cooperative Coevolutionary Cuckoo Search Algorithm

In section, taking inspiration from an organizational evolutionary algorithm, we present a cooperative coevolutionary cuckoo search algorithm (CCCS) which integers annexing operator and cooperating operator, in the core the cuckoo search algorithm. This proposed model will focus on enhancing diversity and the performance of the cuckoo search algorithm.

3.1. Splitting Operator. When a size is too large usually it is split into several small organizations; let Max_{or} be the parameter controlling the maximum size of organization.

3.2. Annexing Operator. Two organizations, $\text{org}_{p1} = \{x_1, x_2, \dots, x_M\}$ and $\text{org}_{p2} = \{y_1, y_2, \dots, y_N\}$, are randomly selected from the current generation. Choose their leaders using CS algorithm. If the org_{p1} is the winner; thus org_{p1} will annex org_{p2} to generate a new organization, $\text{org}_c = \{z_1, z_2, \dots, z_M, z_{M+1}, z_{M+2}, \dots z_{M+N}\}$, where $z_i = x_i$, $i = 1, 2, \dots, M$. Let AS be a predefined parameter. Then, if rand < AS, z_j, $j = M+1, M+2, \dots, M+N$ are generated by (4). Otherwise they are generated by (5). x_k is the leader of an organization and r_{jk} are new member

$$r_{jk} = \begin{cases} \underline{x_k} & \beta_k < \underline{x_k}, \\ \overline{x_k} & \beta_k > \overline{x_k}, \ \beta_k = x_k + \partial_k \times (x_k - y_{jk}), \\ & k = 1, 2, \dots, n, \\ \beta_k & \text{otherwise}, \end{cases} \qquad (4)$$

$$r_{jk} = \begin{cases} \underline{x_k} + \partial \times (\overline{x_k} - \underline{x_k}) & \beta_k(0,1) < \dfrac{1}{n}, \\ x_k & \text{otherwise}. \end{cases} \qquad (5)$$

After r_j, $j = 1, 2, \dots N$ are generated, z_{j+M}, $j = 1, 2, \dots, M$ are determined in (6):

$$z_{j+M} = \begin{cases} r_j & r_j < y_j, \\ r_j & (y_j > r_j), \\ & \{U_j(0,1)\} < \exp(f(r_j) - f(y)), \\ y_j & \text{otherwise}. \end{cases} \qquad (6)$$

3.3. Cooperating Operator. Two organizations, $\text{org}_{p1} = \{x_1, x_2, \dots, x_M\}$ and $\text{org}_{p2} = \{y_1, y_2, \dots, y_N\}$, are randomly selected from the current generation. Let CS $\in (0, 1)$ be a predefined parameter; if rand < CS, the child organization is generated in (7). x_k, y_k are the leader of organization respectively. Otherwise use (8); i is random integer and uses

```
Begin
    Initializing population $p_0$ with $n_0$ organizations, and each organization has one member;
        $t \leftarrow 0$;
    While (the termination criteria are not reached) do
    Begin
        For each organization in $p_t$, if the number of it more than 20, performing the splitting
        operator on it, deleting it from $p_t$, and adding the child organizations into $p_{t+1}$;
        While (the number of organizations in is $p_t$ greater than 1) do
        Begin
            Randomly selecting two parent organizations $org_{p1}$ and $org_{p2}$ from $p_t$;
            Performing the CS and selecting their leaders;
            If rand < 0.5
                Annexing operator;
            Else
                Cooperating operator;
                If $f(w) < f(x)$ and $f(z) < f(y)$
                    $x = w$;
                    $y = z$;
                End
                adding the child organizations into $p_{t+1}$;
            End
            Deleting $org_{p1}$and $org_{p2}$form $p_t$;
        End
        Deleting the $u$ organizations form $p_{t+1}$;
        % $u$ is the child number of join organizations
        $p_t \leftarrow p_{(t+1)}$;
        $t \leftarrow t + 1$;
    End
    output the best solution in $p_t$
End
```

Algorithm 1: A cooperative coevolutionary cuckoo search algorithm (minimum).

flip operator;

$$\begin{gathered} q_k = \partial_k \times x_k + \left(1 - \partial_k\right) \times y_k, \\ r_k = \left(1 - \partial_k\right) \times x_k + \partial_k \times y_k, \\ k = 1, 2, \ldots, n, \end{gathered} \tag{7}$$

$$\begin{tikzcd} x = \left(x_1, x_2, \ldots, \boxed{x_i, \ldots, x_n}\right), \arrow[d] & y = \left(y_1, y_2, \ldots, \boxed{y_i, \ldots, y_n}\right). \arrow[d] \\ w = \left(x_1, x_2, \ldots, \boxed{y_n, \ldots, y_i}\right), & z = \left(y_1, y_2, \ldots, \boxed{x_n, \ldots, x_i}\right). \end{tikzcd} \tag{8}$$

3.4. The Pseudo Code of the Proposed Algorithm is Shown in Algorithm 1. In the initialization, each organization has only one member, and the population has total n_0 organizations. During the evolutionary process, the number of the organizations changes; this is just to maintain the diversity of the population. The main difference between the CCCS and the OEA is that populations changes during the optimization process, in OEA, the number of populations in the optimization process is the same. In contrast, the number of the population in CCCS is changing. In addition, cooperating operators of CCCS and OEA are also different.

4. Problem Definition

A unconstrained optimization problems (UCOPs) are formulated as solving the objective function

$$\text{minimize} \quad f(x), \quad x = \left(x_1, x_2, \ldots, x_n\right) \in s, \tag{9}$$

where $s \subseteq \mathfrak{R}^n$ defines the search space which is an n-dimensional space bounded by the parametric constraints $\underline{x_i} \le x_i \le \overline{x_i}$, $i = 1, 2, \ldots, n$. Thus, $s = [\underline{x}, \overline{x}]$, where $\underline{x} = (\underline{x_1}, \underline{x_2}, \ldots, \underline{x_n})$ and $\overline{x} = (\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$.

A constrained optimization problems (COPs) can be formulated as solving objective function

$$\text{minimize} \quad f(x), \quad x = \left(x_1, x_2, \ldots, x_n\right) \in s \cap \chi, \tag{10}$$

where s is the same with that of (9), and the feasible region χ is

$$\chi = \left\{x \in \mathfrak{R}^n \mid g_j(x) \le 0,\ j = 1, 2, \ldots, m\right\}, \tag{11}$$

where $g_j(x)$, $j = 1, 2, \ldots, m$ are constraints.

4.1. Constraint Handling. In the penalty function approach, nonlinear constraints can be collapsed with the cost function into a response functional. By doing this, the constrained optimization problem is transformed into an unconstrained

Table 1: Experimental result of CCCS on 15 unconstrained benchmark functions over 50 trails.

f	$f_{\min}$	Best function value	Mean function value	Standard deviation	Worst function value
F_{01}	0	$8.804e-242$	$2.011e-217$	0	$2.011e-216$
F_{02}	0	$4.271e-143$	$1.628e-130$	$4.142e-130$	$1.323e-129$
F_{03}	0	$6.280e-192$	$5.464e-154$	$1.729e-153$	$5.464e-153$
F_{04}	0	$1.085e-121$	$1.333e-098$	$4.080e-098$	$1.294e-097$
F_{05}	0	25.0366	25.4182	0.3013	25.8868
F_{06}	0	0	0	0	0
F_{07}	0	$5.309e-005$	$3.080e-004$	$2.391e-004$	$8.047e-004$
F_{08}	−12569.5	$-1.25694870e+004$	$-1.25694867e+004$	$4.830e-004$	$-1.25694860e+004$
F_{09}	0	0	0	0	0
F_{10}	0	$1.655e-7$	$3.663e-5$	$2.654e-9$	$3.188e-4$
F_{11}	0	0	0	0	0
F_{12}	0	$6.845e-011$	$9.518e-007$	$2.780e-006$	$8.635e-006$
F_{13}	0	$2.073e-013$	$1.723e-007$	$2.657e-007$	$6.775e-007$
F_{14}	−99.60	−96.6008	−83.7328	6.1910	−75.6993
F_{15}	−78.33236	−78.33233	−78.33232	$8.654e-006$	−78.33230

Table 2: Comparison between MECA, OEA, and CCCS over 50 trials.

f	$f_{\min}$	Mean function value			Standard deviation		
		CCCS	MECA	OEA	CCCS	MECA	OEA
F_{01}	0	$\mathbf{2.011e-217}$	$4.228e-183$	$2.481e-30$	**0**	0	$1.128e-29$
F_{02}	0	$\mathbf{1.628e-130}$	$1.845e-110$	$2.068e-13$	$\mathbf{4.142e-130}$	$3.113e-110$	$1.440e-12$
F_{03}	0	$\mathbf{5.464e-154}$	$3.274e-95$	$1.883e-9$	$\mathbf{1.729e-153}$	$2.313e-94$	$3.726e-9$
F_{04}	0	$\mathbf{1.333e-098}$	$5.124e-2$	$8.821e-2$	$\mathbf{4.080e-098}$	$9.732e-2$	$2.356e-2$
F_{05}	0	**25.4182**	$7.973e-2$	0.227	**0.3013**	$5.638e-1$	0.941
F_{06}	0	**0**	0	0	**0**	0	0
F_{07}	0	$\mathbf{3.080e-004}$	$4.084e-4$	$3.297e-3$	$\mathbf{2.391e-004}$	$3.800e-4$	$1.096e-3$
F_{08}	−12569.5	**−12569.4867**	−12569.4866	−12569.4866	$\mathbf{4.830e-004}$	$7.350e-12$	$5.555e-12$
F_{09}	0	**0**	0	$5.430e-17$	**0**	0	$1.683e-16$
F_{10}	0	$\mathbf{3.663e-5}$	0	$5.336e-14$	$\mathbf{2.654e-9}$	0	$2.945e-13$
F_{11}	0	**0**	$3.844e-3$	$1.317e-2$	**0**	$7.130e-3$	$1.561e-2$
F_{12}	0	$\mathbf{9.518e-007}$	$1.571e-32$	$9.207e-30$	$\mathbf{2.780e-006}$	$5.529e-48$	$6.436e-31$
F_{13}	0	$\mathbf{1.723e-007}$	$1.350e-32$	$4.323e-19$	$\mathbf{2.657e-007}$	$1.106e-47$	$2.219e-18$
F_{14}	−99.60	**−83.7328**	−98.7094891	−99.5024042	**6.1910**	$1.450e-1$	$2.526e-2$
F_{15}	−78.33236	**−78.33232**	−78.3323314	−78.3323314	$\mathbf{8.654e-6}$	$1.005e-13$	$2.804e-11$

optimization problem simpler to solve [35]. For example, if there are some nonlinear equality constraints ϕ_i and some inequality constraints φ_j, the response functional PI can be defined as follows:

$$\prod \left(x, \mu_i, \nu_j\right) = f(x) + \sum_{i=1}^{M} \mu_i \phi_i^2(x) + \sum_{j=1}^{N} \nu_j \varphi_j^2(x), \quad (12)$$

where $1 \leq \mu_i$ and $0 \leq \nu_i$. The coefficients of penalty terms should be large enough; their values may depend on the specific optimization problem. The contribution of any equality constraints function to the response functional $\prod$ is null but increases significantly as soon as the constraint is violated. The same applies to inequality constraints when they become critical.

5. Implementation in Optimization Problems

All computational experiments are conducted with Matlab R2010a and run on Celeron(R) Dual-core CPU T3100, 1.90 GHZ with 2 GB memory capacity under windows7.

5.1. Experimental Studies on Unconstrained Optimization Problems. In this section, 15 benchmark functions (F_{01}–F_{15}) are used to test the performance of CCCS in solving UCOPS. F_{01}–F_{13} are f_1–f_{13} in [1] and F_{14}, F_{15} are f_7, f_9 in [36]. The problem dimension is set to 30 for F_{01}–F_{13} and 100 for F_{14} and F_{15}. In this manner, these functions have so many local

Table 3: Comparison between MECA, CCGA, CPSO, and CCCS over 50 trials.

f	CCCS	MECA	CCGA	CPSO
F_{05} (f_0)	**25.0366**	$6.43e-1$	3.80	$1.94e-1$
F_{03} (f_1)	**$5.464e-154$**	$1.23e-64$	$1.38e+2$	$2.55e-128$
F_{10} (f_2)	**$1.655e-7$**	0	$9.51e-2$	$2.78e-14$
F_{09} (f_3)	**0**	0	1.22	0
F_{11} (f_4)	**0**	$3.94e-3$	$2.20e-1$	$1.86e-2$

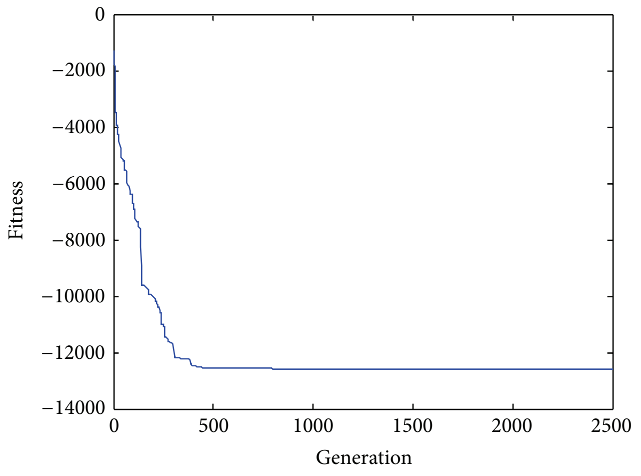

Figure 1: Convergence curve of F_{08}.

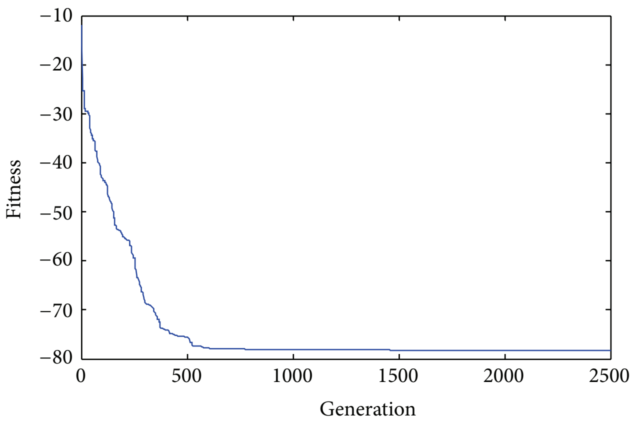

Figure 2: Convergence curve of F_{15}.

minima that they are challenging enough for performance evaluation. The parameters of CCCS are set as follows: the number of organization is 10, others refer to [8], and the number of iterations is 2500.

5.1.1. Experimental Results of CCCS. Table 1 summarizes the experimental results of CCCS, which include the best, the mean, the standard deviation, and the worst function values found. As can be seen, besides F_{05}, F_{14}, other function values reached the theoretical value or are very close to the theoretical value. What should be noted is that the global optimum for F_{11} is 0 every time, but both MECA (M-elite co-evolutionary algorithm for numerical optimization) and OEA (An organizational evolutionary algorithm for numerical optimization) cannot find the global optimal solution. Convergence curve of F_{08} and F_{15} as shown in Figures 1 and 2 respectively, other convergence curve figure has been omitted.

5.1.2. Comparison between CCCS, MECA, and OEA. Table 2 shows statistical results of the CCCS optimization. As can be seen, as for functions F_{01}, F_{02}, F_{03}, F_{04}, F_{07}, F_{08}, and F_{11}, the mean function values of CCCS is better than MECA; as for functions F_{06} and F_{09}, both CCCS and MECA can find the global optimal solution. It is a pity that the results of these functions (F_{05}, F_{10}, F_{12}, F_{13}, and F_{14}) is worse than MECA.

5.1.3. Comparison between MECA, CCGA, CPSO, and CCCS. Table 3 summarizes the experimental results of comparison

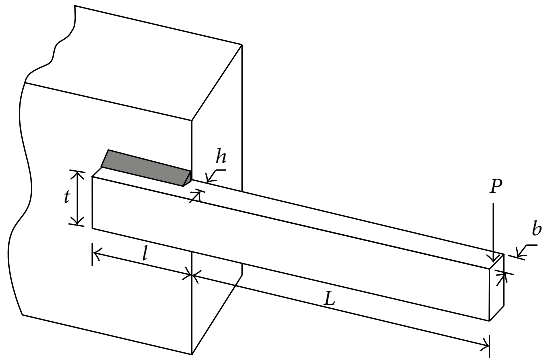

Figure 3: Schematic of the welded beam design problem.

between MECA, CCGA, CPSO, and CCCS. Data of CCGA and CPSO algorithm results from the literature [9], MECA, CCGA, and CPSO requite 200000 function evaluations. But CCCS requires 2500 function evaluations to complete the optimization process. As a whole, the results of CCCS are better than those of MECA, OEA.

5.2. Experimental Studies on Constrained Optimization Problems. In this section, 11 benchmark functions (G_{01}–G_{11}) and 2 engineering design problems (welded beam design, pressure vessel design) are used to validate the performance of CCCS in solving constrained optimization problems. These functions are described in [37]. The equality constraints have been converted into inequality constraints, $|g(x) - \delta| \le 0$, using the degree of violation $\delta = 0.00001$, the same with that of [37] (see Figures 3 and 5).

Table 4: The comparison between OEA, SMES, and CCCS.

f	$f_{\min}$	Method	Best FV	Mean FV	St. dev	Worst FV
G_{01}	−15	**CCCS**	**−15**	**−15**	**0**	**−15**
		OEA CHp	−15	−15	0	−15
		OEA CHc	−15	−14.78	7.637e − 1	−12
		SMES	−15.000	−15.000	0	−15.000
G_{02}	−0.803619	**CCCS**	**−0.7947**	**−0.7826**	**0.0093**	**−0.7694**
		OEA CHp	−0.803605	−0.782518	1.483e − 2	−0.746283
		OEA CHc	−0.803589	−0.782676	1.512e − 2	−0.743149
		SMES	−0.803601	−0.795238	1.67e − 2	−0.751322
G_{03}	−1	**CCCS**	**−1.000**	**−1.000**	**0**	**−1.000**
		OEA CHp	−1.000	−1.000	8.470e − 6	−1.000
		OEA CHc	−1.000	−1.000	1.131e − 5	−1.000
		SMES	−1.000	−1.000	2.09e − 4	−1.000
G_{04}	−30665.539	**CCCS**	**−3.0665539e + 004**	**−3.0665539e + 004**	**3.8348e − 012**	**−3.0665539e + 004**
		OEA CHp	−30665.539	−30665.539	1.735e − 11	−30665.539
		OEA CHc	−30665.539	−30665.539	1.837e − 11	−30665.539
		SMES	−30665.539	−30665.539	0	−30665.539
G_{05}	5126.498	**CCCS**	**5.1264981e + 003**	**5.1264981e + 003**	**9.5869e − 013**	**5.1264981e + 003**
		OEA CHp	5126.497	5127.048	7.071e − 1	5130.051
		OEA CHc	5126.532	5315.975	145.473	5900.26
		SMES	5126.599	5174.492	50.06	5304.167
G_{06}	−6961.814	**CCCS**	**−6.9618139e + 003**	**−6.9618139e + 003**	**9.5869e − 013**	**−6.9618139e + 003**
		OEA CHp	−6961.814	−6961.814	5.512e − 12	−6961.814
		OEA CHc	−6961.814	−6961.814	5.512e − 12	−6961.814
		SMES	−6961.814	−6861.284	1.85	−6952.482
G_{07}	24.306	**CCCS**	**2.4306209e + 001**	**2.4306209e + 001**	**4.2164e − 007**	**2.4306209e + 001**
		OEA CHp	24.308	24.373	7.615e − 2	24.655
		OEA CHc	24.307	24.392	1.178e − 1	24.973
		SMES	24.327	24.475	1.32e − 1	24.843
G_{08}	−0.095825	**CCCS**	**−9.5825041e − 002**	**−9.5825041e − 002**	**0**	**−9.5825041e − 002**
		OEA CHp	−0.095825	−0.095825	0	−0.095825
		OEA CHc	−0.095825	−0.095825	0	−0.095825
		SMES	−0.095825	−0.095825	0	−0.095825
G_{09}	680.630	**CCCS**	**6.8063006e + 002**	**6.8063006e + 002**	**1.1984e − 013**	**6.8063006e + 002**
		OEA CHp	680.630	680.632	1.718e − 3	680.638
		OEA CHc	680.630	680.632	2.163e − 3	680.641
		SMES	680.632	680.643	1.55e − 2	680.719
G_{10}	7049.331	**CCCS**	**7.0492e + 003**	**7.0492e + 003**	**4.7796e − 004**	**7.0492e + 003**
		OEA CHp	7052.236	7219.011	60.737	7326.032
		OEA CHc	7100.030	7231.357	86.409	7469.047
		SMES	7051.903	7253.603	136.02	7638.366
G_{11}	0.750	**CCCS**	**0.7500**	**0.7500**	**0**	**0.7500**
		OEA CHp	0.750	0.750	5.993e − 5	0.750
		OEA CHc	0.750	0.750	1.881e − 7	0.750
		SMES	0.75	0.75	1.52e − 4	0.75

The parameters of CCCS are set as follows: the number of iterations is 2500. However, the others of OEA are 24000. The experimental results of OEA are obtained over 50 independent trials. The running environment is the same as the previously. Table 4 shows the comparison results between OEA, SMES, and CCCS. As can be seen, besides G_{02}, other function values reached the theoretical value or are very close to the theoretical value. On the whole, the result of CCCS is better than those of SMES and CCCS.

Table 5: Welded beam problem: comparison of CS results with the literature.

Researcher(s)	Method	h	l	t	b	Cost	No. of evaluation
Coello [11]	GA	0.2088	3.4205	8.9975	0.2100	1.7483	N.A.
Leite and Topping [12]	GA	0.2489	6.1097	8.2484	0.2485	2.4000	6273
Deb [13]	GA	0.2489	6.1730	8.1789	0.2533	2.4331	320,080
Lemonge and Barbosa [14]	GA	0.2443	6.2117	8.3015	0.2443	2.3816	320,000
Bernardino et al. [15]	AISa-GA	0.2444	6.2183	8.2912	0.2444	2.3812	320,000
Atiqullah and Rao [16]	SAb	0.2471	6.1451	8.2721	0.2495	2.4148	N.A.
Hedar and Fukushima [17]	SA	0.2444	6.2175	8.2915	0.2444	2.3810	N.A.
Liu [18]	SA-GA	0.2231	1.5815	12.8468	0.2245	2.2500	26,466
Hwang and He [19]	SA-DSc	0.2444	6.2158	8.2939	0.2444	2.3811	56,243
Parsopoulos and Vrahatis [20]	PSO	N.A.	N.A.	N.A.	N.A.	1.9220	100,000
He et al. [21]	PSO	0.2444	6.2175	8.2915	0.2444	2.3810	30,000
Zhang et al. [22]	EAd	0.2443	6.2201	8.2940	0.2444	2.3816	28,897
Coello [23]	EA	N.A.	N.A.	N.A.	N.A.	1.8245	N.A.
Lee and Geem [24]	HSe	0.2442	6.2231	8.2915	0.2443	2.381	110,000
Mahdavi et al. [25]	HS	0.2057	3.4705	9.0366	0.2057	1.7248	200,000
Fesanghary et al. [26]	HS-SQP	0.2057	3.4706	9.0368	0.2057	1.7248	90,000
Siddall [27]	RAg	0.2444	6.2819	8.2915	0.2444	2.3815	N.A.
Akhtar et al. [28]	SBMh	0.2407	6.4851	8.2399	0.2497	2.4426	19,259
Ray and Liew [29]	SCAi	0.2444	6.2380	8.2886	0.2446	2.3854	33,095
Montes and Oca [30]	BFOj	0.2536	7.1410	7.1044	0.2536	2.3398	N.A.
Zhang et al. [31]	DEk	0.2444	6.2175	8.2915	0.2444	2.3810	24,000
Gandomi et al. [32]	FA	0.2015	3.562	9.0414	0.2057	1.7312	50,000
Present study	**CCCS**	**0.3312**	**10.0000**	**2.4095**	**0.3456**	**2.1730**	**25,000**

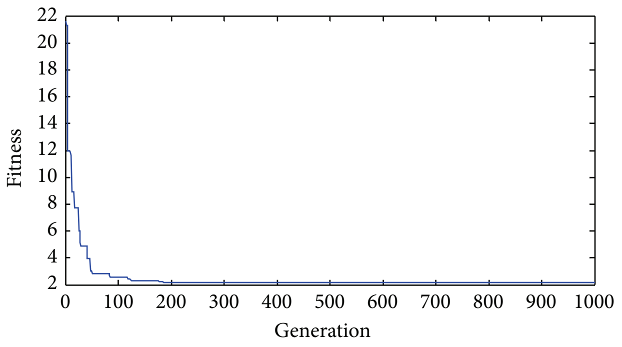

Figure 4: Convergence curve of Case I.

5.3. Implementation in Structural Optimization Problems

Case I (welded beam design). The objective function and parameters of Case I are refers to in [32]. With 25 nests, CCCS found the global optimum requiring 1000 iterations per optimization run. Table 5 compares the optimization results found by CCCS with similar data reported in the literature. CCCS obtained the best design overall of 2.1730. Mahdavi et al. [25] and Fesanghary et al. [26] found a better desgin. "But for the continuous optimization problem equal to 1.7248" is deleted. In addition, CCCS requires only 25,000 function evaluations to complete the optimization process, hence much less than the literature. Convergence curve of Case I as shown in Figure 4.

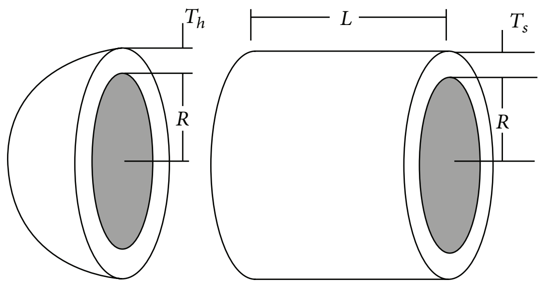

Figure 5: Schematic of the pressure vessel design problem.

Case II (pressure vessel design). The objective function and parameters of the Case II are refers to in [33]. With 25 nests, CCCS found the global optimum requiring 1000 iterations per optimization run. Table 6 compares the optimization results found by CCCS with similar data reported in the literature. CCCS obtained the best design overall of 5885.3. In addition, CCCS requires only 25,000 function evaluations to complete the optimization process, hence much less than the literature. Here, N/A represents no records in the literature. Convergence curve of Case II is shown in Figure 6.

6. Conclusions

Taking inspiration from the OEA, a new numerical optimization algorithm, CCCS, has been proposed in this paper. The experimental results in Tables 1–6 indicate that CCCS

Table 6: Pressure vessel design: comparison of CCCS results with the literature.

Reference method	R	L	T_s	T_h	$f_{\min}$	$\text{iter}_{\max}$
CCCS	**40.3196**	**200.0000**	**0.7782**	**0.3846**	**5885.3**	**1000**
ALPSO [33]	41.35	200	0.798	0.395	6234	7590
PSOA [34]	N/A	N/A	N/A	N/A	6292	6506
PSOSTR [34]	N/A	N/A	N/A	N/A	6272	3723
He et al. [21]	N/A	N/A	N/A	N/A	6290	30000
Akhtar et al. [28]	N/A	N/A	N/A	N/A	6335	12630

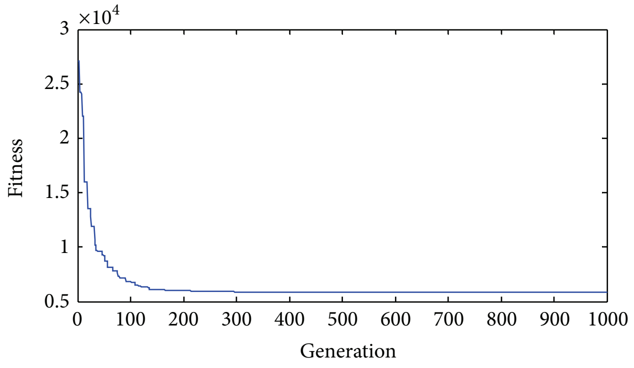

Figure 6: Convergence curve of Case II.

outperforms the MECA and OEA. On the whole, CCCS obtains a good performance for the unconstrained functions, constrained functions, and 2 engineering design problems. These benefits mainly from the following three aspects. One is the dynamics population, and the other is three evolutionary operators. The third aspect is a combination of CS algorithm. 28 experiments illustrate that CCCS has an effective searching mechanism. However, the number of dynamic populations, is difficult to control, which spends lots of computational cost. How to control the number of dynamic population is the future research work.

Acknowledgments

This work is supported by the National Science Foundation of China under Grant no. 61165015, Key Project of Guangxi Science Foundation under Grant no. 2012GXNSFDA053028, Key Project of Guangxi High School Science Foundation under Grant no. 20121ZD008, Open Research Fund Program of Key Lab of Intelligent Perception and Image Understanding of Ministry of Education of China under Grant no. IPIU01201100 and the Innovation Project of Guangxi Graduate Education under Grant no. YCSZ2012063.

References

[1] X. Yao, Y. Liu, and G. M. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[2] Y. W. Leung and Y. P. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 41–53, 2001.

[3] L. Zhang and B. Zhang, "Good point set based genetic algorithm," *Chinese Journal of Computers*, vol. 24, no. 9, pp. 917–922, 2001.

[4] X. S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceeding of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, India, December 2009.

[5] X. S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.

[6] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature—PPSN III*, Y. Davidor, H. P. Schwefel, and R. Mnner, Eds., vol. 866 of *Lecture Notes in Computer Science*, pp. 249–257, Springer, Berlin, Germany, 1994.

[7] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to participle swam optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[8] J. Liu, W. C. Zhong, and L. C. Jiao, "An organizational evolutionary algorithm for numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 37, no. 4, pp. 1052–1064, 2007.

[9] C. Mu, L. Jiao, and Y. Liu, "M-elite coevolutionary algorithm for numerical optimization," *Ruan Jian Xue Bao/Journal of Software*, vol. 20, no. 11, pp. 2925–2938, 2009.

[10] I. Fister, I. Fister Jr., J. Brest, and V. Žumer, "Memetic artificial bee colony algorithm for large-scale global optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, Brisbane, Australia, 2012.

[11] C. A. C. Coello, "Use of a self-adaptive penalty approach for engineering optimization problems," *Computers in Industry*, vol. 41, no. 2, pp. 113–127, 2000.

[12] J. P. B. Leite and B. H. V. Topping, "Improved genetic operators for structural engineering optimization," *Advances in Engineering Software*, vol. 29, no. 7–9, pp. 529–562, 1998.

[13] K. Deb, "Optimal design of a welded beam via genetic algorithms," *AIAA Journal*, vol. 29, no. 11, pp. 2013–2015, 1991.

[14] A. C. C. Lemonge and H. J. C. Barbosa, "An adaptive penalty scheme for genetic algorithms in structural optimization," *International Journal for Numerical Methods in Engineering*, vol. 59, no. 5, pp. 703–736, 2004.

[15] H. S. Bernardino, H. J. C. Barbosa, and A. C. C. Lemonge, "A hybrid genetic algorithm for constrained optimization problems in mechanical engineering," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2007)*, pp. 646–653, Singapore, September 2007.

[16] M. M. Atiqullah and S. S. Rao, "Simulated annealing and parallel processing: an implementation for constrained global design

optimization," *Engineering Optimization*, vol. 32, no. 5, pp. 659–685, 2000.

[17] A. R. Hedar and M. Fukushima, "Derivative-free filter simulated annealing method for constrained continuous global optimization," *Journal of Global Optimization*, vol. 35, no. 4, pp. 521–649, 2006.

[18] J. L. Liu, "Novel orthogonal simulated annealing with fractional factorial analysis to solve global optimization problems," *Engineering Optimization*, vol. 37, no. 5, pp. 499–519, 2005.

[19] S. F. Hwang and R. S. He, "A hybrid real-parameter genetic algorithm for function optimization," *Advanced Engineering Informatics*, vol. 20, no. 1, pp. 7–21, 2006.

[20] K. E. Parsopoulos and M. N. Vrahatis, "Unified particle swarm optimization for solving constrained engineering optimization problems," in *Advances in Natural Computation*, vol. 3612 of *Lecture Notes in Computer Science*, pp. 582–591, 2005.

[21] S. He, E. Prempain, and Q. H. Wu, "An improved particle swarm optimizer for mechanical design optimization problems," *Engineering Optimization*, vol. 36, no. 5, pp. 585–605, 2004.

[22] J. Zhang, C. Liang, Y. Huang, J. Wu, and S. Yang, "An effective multiagent evolutionary algorithm integrating a novel roulette inversion operator for engineering optimization," *Applied Mathematics and Computation*, vol. 211, no. 2, pp. 392–416, 2009.

[23] C. A. C. Coello, "Constraint-handling using an evolutionary multiobjective optimization technique," *Civil Engineering and Environmental Systems*, vol. 17, no. 4, pp. 319–346, 2000.

[24] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 36–38, pp. 3902–3933, 2005.

[25] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied Mathematics and Computation*, vol. 188, no. 2, pp. 1567–1579, 2007.

[26] M. Fesanghary, M. Mahdavi, M. Minary-Jolandan, and Y. Alizadeh, "Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 197, no. 33–40, pp. 3080–3091, 2008.

[27] J. N. Siddall, *Analytical Decision-Making in Engineering Design*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1972.

[28] S. Akhtar, K. Tai, and T. Ray, "A socio-behavioral simulation model for engineering design optimization," *Engineering Optimization*, vol. 34, no. 4, pp. 341–354, 2002.

[29] T. Ray and K. M. Liew, "Society and civilization: an optimization algorithm based on the simulation of social behavior," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 4, pp. 386–396, 2003.

[30] E. M. Montes and B. H. Oca, "Modified bacterial foraging optimization for engineering design," in *Proceedings of the Intelligent Engineering Systems through Artificial Neural Networks (ANNIE '2009)*, C. H. Dagli, K. M. Bryden, and S. M. Corns, Eds., vol. 19 of *ASME Press Series*, pp. 357–364.

[31] M. Zhang, W. Luo, and X. Wang, "Differential evolution with dynamic stochastic selection for constrained optimization," *Information Sciences*, vol. 178, no. 15, pp. 3043–3074, 2008.

[32] A. H. Gandomi, X. S. Yang, and A. H. Alavi, "Mixed variable structural optimization using Firefly algorithm," *Computers and Structures*, vol. 89, no. 23-24, pp. 2325–2336, 2011.

[33] Y. Yu, X. Yu, and Y. Li, "Solving engineering optimization problem by Augmented Lagrange particle swarm optimization," *Journal of Mechanical Engineering*, vol. 45, no. 12, pp. 167–172, 2009.

[34] G. G. Dimopoulos, "Mixed-variable engineering optimization based on evolutionary and social metaphors," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 4–6, pp. 803–817, 2007.

[35] X. S. Yang, *Engineering Optimization: An Introduction with Metaheuristic Applications*, John Wiley & Sons, New York, NY, USA, 2010.

[36] Y. W. Leung and Y. P. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 41–53, 2001.

[37] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 284–294, 2000.