International Journal of
Mathematical Modelling & Computations
Vol. 08, No. 02, Spring 2018, 67- 72

Multiobjective Imperialist Competitive Evolutionary Algorithm for Solving Nonlinear Constrained Programming Problems

C. Liu*

School of Mathematics and Information Sciences, Baoji University of Arts and Sciences, Baoji, China.

Abstract. Nonlinear constrained programing problem (NCPP) has been arisen in diverse range of sciences such as portfolio, economic management etc.. In this paper, a multiobjective imperialist competitive evolutionary algorithm for solving NCPP is proposed. Firstly, we transform the NCPP into a biobjective optimization problem. Secondly, in order to improve the diversity of evolution country swarm, and help the evolution country swarm to approach or land in the feasible region of the problem, three kinds of different methods of colonies moving toward their relevant imperialist are given. Thirdly, the new operator for exchanging position of the imperialist and colony is given similar as a recombination operator in genetic algorithm to enrich the exploration and exploitation abilities of the proposed algorithm. At last, the new approach is tested on two well-known NP-hard nonlinear constrained optimization functions, and the empirical evidence suggests that the proposed method is robust, efficient, and generic.

Received: 03 September 2018, Revised: 03 December 2018, Accepted: 23 February 2019.

Keywords: Multiobjective optimization; Imperialist competitive evolutionary algorithm; Nonlinear constrained optimization; Optimal solution.

1. Introduction

In portfolio and engineering disciplines, many optimization problems involve in the constraint environments [1,3]. That's to say, the optimal solution of those practical problems are restricted to the problem's constraint conditions. When solving these optimization problems, it is difficult to deal with the constraints and find the real optimal solution or approximation optimal solution.

Mostly often, constraint handling methods can be identified two types: one is generic methods that do not exploit the mathematical structure of the constraints, and the other is special methods that used to solve these problems with specific types of constraints. In fact, among the generic methods, the most popular approach is the penalty function method, in which involves a number of penalty parameters and we must to set right in any algorithms in order to obtain the optimal solution.

Imperialist competitive algorithm (ICA), similar to the genetic algorithm, is a kind of generic algorithm proposed by E. Atashpaz-Gargari and C. Lucas [4] in 2007. Imperialist competitive algorithm mainly mimics the social–political process of imperialism and imperialistic competition, it has also been succeeded widely to solve many real world optimization problems in recent years, e.g., in [5], Mahdi A., Ayaz I., and Davoud A. introduced an imperialist competitive algorithm for solving systems of nonlinear

*Corresponding author. Email: liu2006@126.com equations, and in [6], the authors designed a multi-objective imperialist competitive algorithm to solve a capacitated hub covering location problem, E. Shokrollahpour, M. Zandieh and B. Dorri [7] proposed a novel imperialist competitive algorithm for solving bi-criteria scheduling of the assembly flow-shop problem, etc.

In this paper, we proposed a new multiobjective optimization method based on the imperialist competitive algorithm to solve NCOP, Firstly, the considering nonlinear constrained optimization problem is transformed into a bi-objective optimization problem. Then, in order to improve the diversity of evolution country swarm, and help the evolution country swarm to approach or land in the feasible region, three kinds of different methods of colonies moving toward their relevant imperialist are presented. Also, the new operator for exchanging position of the imperialist and colony is given. At last, the new method is tested on three NP-hard nonlinear constrained optimization functions, and compared with four state-of-the-art algorithms, the proposed algorithm has remarkably advantage in terms of the best, mean, the worst objective function values and the standard deviations, i.e., it is indicated that the proposed algorithm can effectively solve NCOP.

2. Related concepts of NCOP

Considered the following nonlinear constrained optimization problem (NCOP):

$$\begin{cases} \min\limits_{x\in D\subseteq[L,U]} f(x) \\ s.t.\ g_i(x)\le 0,\ i=1,2,\cdots,p, \\ \quad h_j(x)=0,\ j=p+1,p+2,\cdots,l \end{cases} \tag{1}$$

where $x=(x_1,x_2,\cdots,x_n)^T$ is the n dimension decision vector and $g_i(x)\ge 0$ is the inequality constraint for $i=1,2,\cdots,p.$ $h_j(x)=0$ is equality constraint for $j=p+1,p+2,\cdots,l$, and

$$D=\{x|g_i(x)\le 0, i=1,2,\cdots,p; h_j(x)=0, j=p+1,p+2,\cdots,l\} \tag{2}$$

is called the feasible region, and $[L,U]$ is called the search region.

Definition 1. For every point $x\in D$, if exists a point $x^*\in D$ such that $f(x^*)\le f(x)$ holds, then the point x^* is called the optimal solution, and $f(x^*)$ is the optimal value for problem (1).

Let $f_1(x)=f(x)$ and $f_2(x)=\max\limits_{1\le i\le p}\{0,g_i(x)\}+\sum_{j=p+1}^{l}(h_j(x))^2$, where $f(x)$ is the objective function of problem (1) and $f_2(x)$ is the optimization function defined by the constraints condition of problem (1), then, we can transform the nonlinear constrained optimization problem (1) into the biobjective optimization problem as follows:

$$\min_{x\in D\subseteq[L,U]} F(x)=(f_1(x),f_2(x)) \tag{3}$$

For the biobjective optimization problem (3), when to minimize the first objective function $f_1(x)$, it means to find a feasible point so as to become the optimal solution of problem (1), and to minimize the second objective function $f_2(x)$, it means to search the point in order to meet all the constraints of problem (1). Therefore, when to minimize the two objectives function of problem (3) simultaneously means searching for the point so as to satisfy all the constraints and make the objective function of problem (1) to reach the optimum.

3. Imperialist competitive algorithm

Imperialist competitive algorithm (ICA) is proposed by the authors E. Atashpaz-Gargari and C. Lucas [4] in 2007, during the operation process of ICA, the initial evolution country swarm should be generated firstly. Among the initial country swarm, some of the best countries are selected to form the initial imperialists, and the rest of the countries are divided among the initial imperialists as colonies. Then, each imperialist along with its colonies is regard as empire. All the empires start to compete among each other. The weakest empire, which cannot succeed in this competition, will be eliminated from the competition and take part in the strongest power imperialist. Finally, the collapse mechanism will cause all the colonies to converge to a state which is the optimal solution.

Initial empires creation. Randomly generate pop initial countries in search space, denote them as $\text{country}i = (x_1^i, x_2^i, \cdots, x_n^i)^T$ for $i = 1, 2, \cdots, pop$, and define the cost of each $\text{country}i$ as follows:

$$\text{cost}(\text{country}i) = \begin{cases} f_1(\text{country}i) & \text{country}i \in D \\ f_2(\text{country}i) & \text{country}i \in [L, U] \setminus D \end{cases} \tag{4}$$

Select N of the most powerful countries to form empires, the rest countries of the initial countries will become colonies of each of empires according to their power. Thus, each empire receives a number of colonies. At last, these initial countries is divided into two groups: imperialist and colony (denote in imperialist*i* and colony*j* for $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, pop - N$, respectively). In order to form the initial empires, we divide colonies into N imperialists based on their power. Here, we divide these colonies among imperialists according to the roulette wheel selection as follows:

*Step1***:** Suppose the normalized power of each imperialist is defined by

$$p_j = \left| \frac{C_j}{\sum_{i=1}^{N} C_i} \right|, \tag{5}$$

where p_j is the normalized power of the *j*-th imperialist, and $C_j = c_j - \max_{1 \le i \le N}\{c_i\}$ is the normalized cost of the *j*-th imperialist*j* for $j = 1, 2, \cdots, pop - N$ is the cost of the *i*-th imperialist*i* for $i = 1, 2, \cdots, N$.

Step2: Generate the initial number of the colonies belonging to each empire based on the following formula

$$N.C._j = round\{p_j \cdot (pop - N)\}.$$

where $N.C._j$ is the number of initial colonies of the *j*-th empire, and *pop-N* is the total number of all initial colonies.

Step3: Select $N.C._j$ of the colonies based on the roulette wheel selection and join them to the *j*-th imperialist. These colonies along with the imperialist together will form the *j*-th empire (denotes empire*j*, $j = 1, 2, \cdots, pop - N$).

Method of colonies moving toward their relevant imperialist. In [4], the authors make each colony to move toward the imperialist by *x*-units in the direction which is the vector from colony to imperialist. *x* will be a random variable with uniform distribution, i.e.,

$$x \sim U(0, \beta \times d) \tag{6}$$

where $\beta > 1$ and *d* is the distance between the colony and imperialist, parameter β causes the colony to get closer to the imperialist from both sides. In this section, we proposed a new method of colonies moving to their relevant imperialist based on the character of optimization problem (1). Suppose that we make the colony*j* ($j = 1, 2, \cdots, pop - N$) to move the imperialist*i*($i = 1, 2, \cdots, N$), then,

Case1: If both imperialist*i* and colony*j* are feasible, then we generate a circle which the diameter is the straight-line segment *d* joining the imperialist*i* and colony*j* , the new position colony*k* of colony*j* moved is shown in a gray colour in Figure1(a), where *A* and *r* are two random numbers with uniform distribution, i.e.,

$$A \sim U(-\pi, \pi) \text{ and } r \sim U(0, d \cdot \cos A). \tag{7}$$

Case2: If both imperialist*i* and colony*j* are infeasible, then, we random select a colony*s* from feasible region *D*, then compute the barycenter of three countries colony*j*, imperialist*i* and colony*s*, then, the barycenter (denoted by colony*k* in Figure1(b)) can be regarded as the new position which colony*j* move to imperialist*i*.

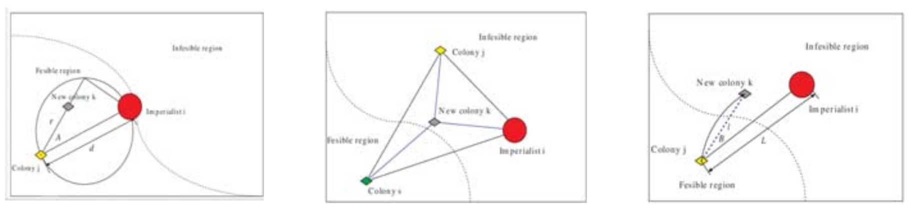
Case3: If exists one feasible country between colony*j* and imperialist*i*, suppose colony*j* is feasible country and imperialist*i* is infeasible country and vice versa, then, we generate a circle which the circle's center is colony*j* and the radius is the straight-line segment *L* joining the colony*j* and imperialist*i*, the new position colony*k* of colony*j* moved to imperialist*i* is shown in a gray colour in Fig.1(c), where *B* is a random number with uniform distribution, i.e., $B \sim U(-\varphi, \varphi)$, where φ is a parameter that adjusts the deviation of direction which is the vector from colony*j* to imperialist*i*, $l \sim U(o, \tau \cdot L)$ is a random number with uniform distribution, and τ and φ are arbitrary. In most of our implementation, the value of $\tau < 1/2$ and $\varphi = 1/\pi$ have a good convergence to the global minimum and can make the feasible colony*j* not far away from the feasible region.

Exchanging position of the imperialist and colony. The operator of exchanging positions of the imperialist and the colony can be described as follows:

(1) If both colony *j* and imperialist *i* are feasible, and suppose that the cost of colony *j* has lower cost than that the imperialist *i* does, i.e., f_1(colony *j*)<f_1(imperialist *i*), then we use the colony *j* to replace the imperialist *i* and form the new imperialist, vice versa.

(2) If both colony *j* and imperialist *i* are unfeasible, then we always choose the one with the smaller cost as the new imperialist *i*, i.e., if f_2(colony*j*)>f_2(imperialist*i*), then keep imperialist*i* invariable, otherwise, if f_2(colony*j*)<f_2(imperialist*i*), then use the colony *j* to replace the imperialist*i* and form new imperialist.

(3) If exists one feasible country between the colony*j* and imperialist*i*, we always use the feasible as the new imperialist in order to make the evolution country swarm approaching the feasible region and fast convergence to the minimum.



(a) (b) (c)

Figure 1. The method of colony moving to imperialist based on colony*j* and imperialist*i* in search space [*L*,*U*]; (a) both colony*j* and imperialist*i* are feasible; (b) both colony*j* and imperialist*i* are infeasible; (c) colony*j* is feasible and imperialist*i* is infeasible.

4. Multiobjective imperialist competitive evolutionary algorithm

Step1 Generate initial country size *pop*, randomly generate initial country swarm in search space [*L*,*U*], and denotes them as the set *pop*(0), find the weekly Pareto optimal countries in *pop*(0) and denote them as an external set *C*(0), let *t*=0.

Step2 Generate initial empires, i.e. select *N* of the most powerful countries from *pop*(*t*) and divide the rest countries to each of them.

Step3 Make each of colonies to move toward relative imperialist based on the method of colonies moving toward their relevant imperialist, and exchange the position of the imperialist and the colony.

Step4 Find the weekly Pareto optimal countries in the set $C(t) \cup pop(t+1)$ and use them to replace those countries in *C*(*t*) to form the new external set *C*(*t*+1).

Step5 If the maximum number of the cycles has been reached, the algorithm stop; output the optimal solution of problem (1), otherwise, let *t*=*t*+1, go to step 2.

5. Numerical simulation

Test problems. To evaluate the efficiency of the proposed algorithm, two nonlinear constrained optimization test problems were tested by five optimization evolutionary algorithms: SAEA [8], SMEA [9], RCEA [10] and ISEA [11] and the proposed algorithm MICA. The two benchmark functions are described in [10].

$$\text{g01.}\ \begin{cases} \max\ f(x) = \left| \dfrac{\sum_{i=1}^{n} \cos^4 x_i - 2\prod_{i=1}^{n} \cos^2 x_i}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right|, \\ \text{s.t.}\ \ g_1(x) = 0.75 - \prod_{i=1}^{n} x_i \le 0, \\ \qquad g_2(x) = \sum_{i=1}^{n} x_i - 7.5n \le 0, \end{cases}$$

where n=20, $0 \le x_i \le 10\ (i = 1\cdots n)$, the global maximum is unknown, *f*(*x**)=-0.803619 is better than any reported value up to the best of our knowledge, and the constraint g_1 is active.

$$\text{g02.}\ \begin{cases} \min\ f(x) = (x_1-10)^3 + (x_2-20)^3 \\ s.t.\ \ g_1(x) = -(x_1-5)^2 - (x_2-5)^2 + 100 \le 0, \ . \\ \qquad g_2(x) = (x_1-6)^2 - (x_2-5)^2 - 82.81 \le 0, \end{cases}$$

The best known solution $x^* = (14.095,\ 0.84296)$, and *f*(*x**)=-6961.81388.

Results analysis. We list the known optimal solution and the best, mean and worst objective function values in Table 1, and the standard deviations after 30 independent runs by MICA is also given. These results provided by four compared algorithms SAEA, SMEA, RCEA and ISEA were taken from the original references.

Table 1. The results obtained by SAEA, SMEA, RCEA, ISEA and MICA for g01and g02.

Function	Optimal	Status	Methods				
			SAEA[8]	SMEA[9]	RCEA[10]	ISEA[11]	MICA
g01	-0.803619	best	-0.80297	-0.803601	-0.803515	-0.803376	-0.803619
		mean	-0.79010	-0.785238	-0.781975	-0.798231	-0.793421
		worst	-0.76043	-0.751322	-0.726288	-0.768291	-0.783461
		st.dev	1.2E-02	1.7E-02	2.0E-02	9.0E-03	2.5E-02
g02	-6961.8138	best	-6961.800	-6961.814	-6961.814	-6961.814	-6961.814
		mean	-6961.800	-6961.284	-6875.940	-6961.813	-6961.814
		worst	-6961.800	-6952.482	-6350.262	-6961.812	-6961.814
		st.dev	0.0000	1.9E+00	1.6E+02	8.5E-05	1.21E-10

It can be seen from Table 1, compared with the four algorithms SAEA, SMEA, RCEA and ISEA, our algorithm MICA can find a better ``best'' result in functions g01. In addition, algorithm MICA found a ``similar'' best solution in problems g02 than SMEA, RCEA and ISEA do. Our approach found better ``mean'' and ``worst'' results in test functions g02 than the compared algorithms. Thus, it reflects the fact that our algorithm is capable of performing a robust and stable search.

6. Conclusion

This paper introduce a new imperialist competitive algorithm (MICA) for solving nonlinear constrained optimization problem. From the comparative study, algorithm NICA has shown its potential to handle various nonlinear constrained optimization problems. This work is partially supported by The Planning fund for the humanities and social sciences of the Ministry of Education (No.18YJA790053).

References

[1] A. Homaifar, S. H. Y. Lai and X. Qi, Constrained optimization via genetic algorithms, Simulation, **62 (4)** (1994) 242-254.

[2] C. A. C. Coello, Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art, Comput. Methods Appl. Mech. Eng, **191 (11/12)** (2002) 1245-1287.

[3] S. B. Hamida and M. Schoenauer. ASCHEA: new results using adaptive segregational constraint handling, in Proc. Congr. Evol. Comput, IEEE Press, (2002) 56-78.

[4] E. Atashpaz-Gargari and C. Lucas, Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition, In: IEEE Congress on Evolutionary Computation, (2007) 4661-4667.

[5] M. Abdollahi, A. Isazadeh and D. Abdollahi, Imperialist competitive algorithm for solving systems of nonlinear equations, Computers and Mathematics with Applications, **65** (2013) 1894-1908.

[6] M. Mohammadi, R. Tavakkoli-Moghaddam and H. Rostamib, A multi-objective imperialist competitive algorithm for a capacitated hub covering location problem, International Journal of Industrial Engineering Computations, **2** (2011) 671-688.

[7] E. Shokrollahpour, M. Zandieh and Behrouz Dorri, A novel imperialist competitive algorithm for bi-criteria scheduling of the assembly flowshop problem, International Journal of Production Research, **49 (11)** (2011) 3087-3103.

[8] R. Farmani and J. A. Wright, Self-adaptive fitness formulation for constrained optimization, IEEE Trans. Evol. Comput, **7 (5)** (2003) 445-455.

[9] E. Mezura-Montes and C. A. C. Coello, A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Trans. Evol. Comput, **9 (1)** (2005) 1-17.

[10] T. P. Runarsson and X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Trans. Evol. Comput, **4 (3)** (2000) 284-294.

[11] A. H. Aguirre, S. B. Rionda, C. A. C. Coello, G. L. Lizaraga, and E. Mezura-Montes, Handling constraints using multiobjective optimization concepts, Int. J. Numerical Methods in Eng, **59 (15)** (2004) 1989-2017.