2nd International Conference on Engineering Optimization
September 6 - 9, 2010, Lisbon, Portugal

Imperialist Competitive Algorithm for truss structures with discrete variables

Hadi Eskandar [1], Pouya Salehi [2], A. Ghoddosian [3]

[1] University of Semnan, Semnan, Iran, hadi.eskandar@yahoo.com
[2] University of Semnan, Semnan, Iran, pouya.salehi@gmail.com

Abstract
Although the goal of optimization is to reduce the costs of a structure construction, but it is clear the considering of economical conditions and engineering criterions. This study is a new method called Imperialist Competitive Algorithm (ICA) which is employed to optimization of several truss structures with discrete variables, and compared with the Heuristic Particle Swarm Optimization (HPSO) and other methods. The results show that the ICA is the robust algorithm for finding the global optimum and has the fastest convergence rate among these methods. Also by the solved examples, it is shown that this method is quite proper even for constrained problems. Furthermore, the proposed ICA can be effectively used to solve optimization problems for structures with discrete variables.
Keywords: Optimization; Truss structures; Assimilation; Imperialistic competition; Discrete variables

1. Introduction

In recent decades, different optimizing algorithms for truss optimization are widely used, and this makes the designing of truss structure optimizers to be attractive for researchers in the optimization field. There are three main categories in structural optimization:
a) Sizing Optimization (the cross-sectional areas of the members are considered as design variables [1, 2]).
b) Shape Optimization (The nodes coordinates are considered as the design variables [2]).
c) Topology Optimization (The location of links in which connect the nodes to each other, are considered as design variables [3]). While optimizing a problem, we may consider two or all of these three types, at the same time.
The design variables in most of optimization problems are continuous. However, in reality in some problems, the design variables such as cross-sectional areas have discrete amounts due to production of bars with special cross sectional areas. The discrete optimization problems are being solved by using mathematical methods or round-off techniques for continuous solutions. The solutions calculated by this method, may be infeasible or be far from optimum solution [4].
Recently, the new methods such as genetic algorithm [5], simulated annealing (SA) [6], particle swarm optimization (PSO) [7] and other stochastic searching methods (which are inspired from the nature) are used in optimizing the trusses.
This paper presents an imperialist competitive algorithm (ICA) that is one of the newest algorithms in optimization field. This algorithm suggested by Atashpaz et al… [8-9]. The ICA method is inspired from a social-Human phenomenon and has two great characteristics; a) high ability of this algorithm to search the global optimization even when facing with nonlinear optimization problems and b) fast convergence speed [9]. In this paper, the ICA method is applied to the structural optimization problems. The work shows the proposed ICA can be effectively used to solve optimization problems for steel structures with discrete variables.
The present paper is organized as follows: After present introductory section in Section 2, the Imperialist Competitive Algorithm is described. The formula for discrete optimizing problems is driven in Section 3. Various examples are studied in Section 4 and the advantages of the ICA are discussed. At the end in Section 5, the conclusions are presented.

2. The Imperialist Competitive Algorithm

Imperialist Competitive Algorithm is inspired from the socio-political process of imperialism and imperialistic competition. This algorithm (like many optimization algorithms) starts with an initial population. Each individual of the population is called a ‘country’. Some of the best countries with the minimum cost are considered as the imperialist states and the rest will be the colonies of those imperialist states. All the colonies are distributed among the imperialist countries regarding their power. The power of each country is inversely proportional to its cost, which is the fitness value in the GA.

2.1. Creation of initial empires

Since our final goal in optimization is to find an optimal solution in terms of the variables of the problem, an array of variable that are going to be optimized should be formed. When solving a N_{var} dimensional optimization problem, a country is a $1 \times N_{var}$ array. This country is defined as follow:

$$Country=[p_1,p_2,p_3,\ldots,p_{Nvar}] \tag{1}$$

where p_is are considered as the variables that should be optimized. when the problem was optimized, the optimal solution is going to be find which is the one with the minimum cost value. By evaluating the cost function, f, for variables (p_1, p_2, p_3,…,p_{Nvar}), the cost of a country will be found (Equation (2)):

$$c_i = Cost_i = f(country_i) = f(p_1,p_2,p_3,...,p_{N_{var}}) \tag{2}$$

To define the algorithm, first of all, initial countries of size $N_{Country}$ is produced. Then, we select N_{imp} of the most powerful countries to form the empires. Therefore, the rest with the size N_{col} will form the colonies that belong to an empire. Then, the colonies are divided among imperialists according to their power. In order to do that, the normalized cost of an imperialist is defined by [9]:

$$C_n = \max_i\{c_i\} - c_n \tag{3}$$

In the above equation, cn is the cost of nth imperialist and Cn is its normalized cost. When the normalized costs of all imperialists are gather, the normalized power of each imperialist is evaluate according to the following equation:

$$P_n = \left| \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \right| \tag{4}$$

As mentioned before, the initial colonies are distributed among empires based on their power and the initial number of the colonies of the nth empire will be obtained from following equation:

$$N.C_n = round\{P_n.N_{col}\} \tag{5}$$

In the above equation, $N.C._n$ is the initial number of colonies of the nth empire and N_{col} is the total number of initial colonies. To divide the colonies, $N.C._n$ of the colonies are randomly chosen and given to the nth imperialist. These colonies along with the nth imperialist form the nth empire [9].

2.2. Assimilation: giving a move to colonies toward the imperialist

According to assimilation policy, the imperialist states tend to draw their colonies toward themselves and make them a part of themselves. The assimilation policy in the ICA is illustrated by giving a move to all the colonies toward the imperialists. In figure (1), you can see such a movement in which a colony moves toward the imperialist by x units. Where, x is a random variable with uniform (or any proper) distribution [8].

$$x \sim U(0, \beta \times d), \beta > 1 \tag{6}$$

In the above equation, β is a number greater than one and d is the distance between the colony and the imperialist. This fact, β>1 causes the colony to get closer to the imperialist from both sides [8].

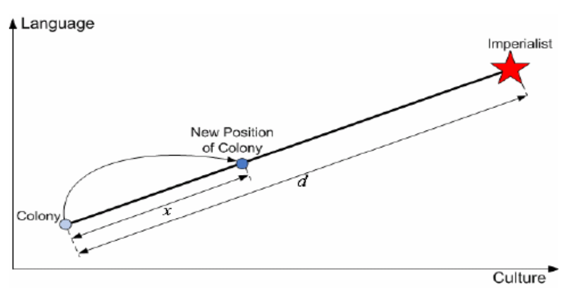

Figure 1- giving a move to the colonies toward their corresponding imperialist [9]

When the imperialist state assimilates the colony, the direction of the movement will not necessarily be a straight line from colony to the imperialist. In order to illustrate this fact and to increase the ability of searching more area around the imperialist, a random amount of deviation is added to the direction of movement. Figure (2), shows the new direction. In this figure, θ is a parameter with uniform (or any proper) distribution [8].

$$\theta \sim U(-\gamma.\gamma) \tag{7}$$

In the above equation, γ is considered as a parameter that adjusts the deviation from the initial direction.

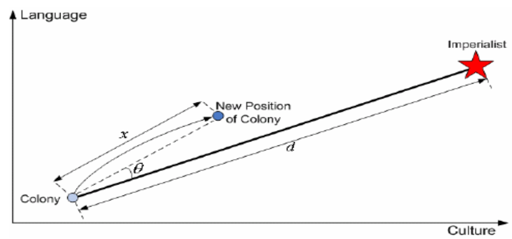

Figure 2- giving a move to the colonies toward their corresponding imperialist in an accidental deviated orientation

2.3. Exchanging positions of the imperialist and a colony
During any movement, if a colony reaches a better point than an imperialist, they will be replaced by each other. After that, the algorithm will continue by the imperialist in the new position and the colonies will be assimilated by the imperialist in its new position. [9].

2.4. Total power of an empire
The total power of an empire is obtained by:

$$\mathrm{TC}_n = \mathrm{Cost}\,(\mathrm{imperialist}_n) + \xi\,\{\mathrm{mean}\,(\mathrm{Cost}(\mathrm{colonies\ of\ empire}_n))\} \tag{8}$$

In the above equation, TCn is the total power of the nth empire and ξ is a small positive number. A small amount for ξ should be chosen to make the cost of an imperialist more important than the cost of colonies for estimating the total power of an empire. The value of 0.1 for ξ has shown good results in most of the previous works [9].

2.5. Imperialistic competitions
There has always been a competition among the empires to take control and possess each other's colonies. During this competition usually the weakest colony (or colonies) of the weakest empire is chosen by other empires and the competition is started on possessing this colony. In Figure (3), a competition is illustrated. Each imperialist participating in this competition, has a chance of possessing the cited colony, according to its power [9].

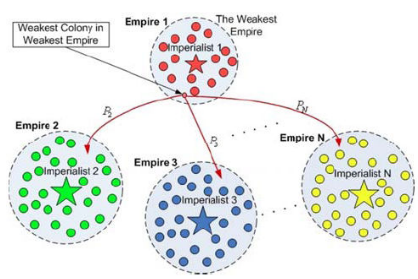

Figure 3- Imperialistic Competitions [9]

To start the competition, first of all, the weakest empire is chosen and then the possession probability of each empire is estimated. The possession probability P_p is related to the total power of the empire. In order to evaluate the normalized total cost of an empire, the following equation is used [9]:

$$NTC_n = \max_i \{TC_i\} - TC_n \tag{9}$$

In the above equation, TCn is the total power of the nth empire and NTCn is the normalized total power of nth empire. When the normalized total power is obtain, To estimate the possession probability of each empire the following equation is used:

$$P_{P_n} = \left| \frac{N.T.C_n}{\sum_{i=1}^{N_{imp}} N.T.C_i} \right| \tag{10}$$

Note that in this way the powerful empires have more chance in possessing the weakest colony of the weakest empire.
In order to divide the given colonies among the empires, vector P is formed as follows:

$$P = [P_{P_1}, P_{P_2}, P_{P_3}, ..., P_{P_{N_{imp}}}] \tag{11}$$

After that, the vector R should be defined with the same size of vector P. The elements of vector R are random numbers between 0 and 1.

$$R = [r_1, r_2, r_3, ..., r_{N_{imp}}] \quad , r_1, r_2, r_3, ..., r_{N_{imp}} \sim U(0,1) \tag{12}$$

Then, vector D is constructed by subtracting R from P.

$$D = P - R = [D_1, D_2, D_3, ..., D_{N_{imp}}] = [P_{P_1} - r_1, P_{P2} - r_2, P_{P3} - r_3, ..., P_{P_{N_{imp}}} - r_{N_{imp}}] \tag{13}$$

When an empire achieves the maximum related index in D, it can take control of the given colony.
During the imperialistic competition, the weak empires will slowly lose their power and getting weakened by the time. they will vanish and at the end, there will remain just one empire that govern the whole colonies [9].

2.6. Stopping Conditions
In most of the optimization methods, including the method which has been explained in this work, the criterion of stopping the algorithm can be the given maximum iteration number, or in some continuous generations, the time in which the amount of the objective function has no improvement, etc. In this method, in addition to the mentioned criteria, when only one empire remains, is also considered as the stopping condition.

2.7. Imposing constraints
Two methods are suggested for applying the constraints :
a) Converting the constrained problem to an unconstrained problem, by using the penalty function which is the most common method. By using this method the objective function and the constraints, are transformed into the fallowing form:

$$\Phi(X, r_p) = F(X) + r_p P(X) \tag{14}$$

$$P(X) = \sum_{j=1}^{m} \{\max[\ 0, g_j(X)]\}^2 \tag{15}$$

In the above equations, Φ(X, rp) is the new objective function, F(X) is the initial objective function, rp is a positive penalty parameter, gj(X) are the constraints applied to the problem and m is the number of the constraints.

b)Deletion of infeasible solution directly. In this method, there are four rules [10]:

Rule 1: Any feasible solution is preferred to any infeasible solution.

Rule 2: Infeasible solutions including slight violation of the constraints (from 0.01 in the first iteration to 0.001 in the last iteration) are considered as feasible solutions.

Rule 3: Between two feasible solutions, the one that have the lower objective function value is preferred.

Rule 4: Between two infeasible solutions, the one that have the lower sum of constraint violation is preferred.

Since in recent works the second approach is used widely (such as HPSACO) and capability of this approach to find the global optimum is better than the first one, in this work also the second approach is employed.

2.8. Algorithm of the ICA

Step one: initializing the empire by selecting some random points. Selection of the best countries as the imperialists is based on their cost function values and satisfaction of constraints.

Step two: giving a move to the colonies toward their imperialist (Assimilation)

Step three: If in an empire, a colony has lesser costs than an imperialist and also the constraints are satisfied, position of the imperialist and a colony should be changed.

Step four: Estimating the total cost of all empires

Step five: taking possession of the weakest colony (colonies) of the weakest empire by another empire that has the most chance of possessing it (them). (Imperialistic competition)

Step six: if an empire doesn't have any colonies, eliminate this empire.

Step seven: stop the process when stopping conditions are satisfied. Otherwise go to the 2nd step.

The flowchart of Imperialist Competitive Algorithm is illustrated in figure (4).

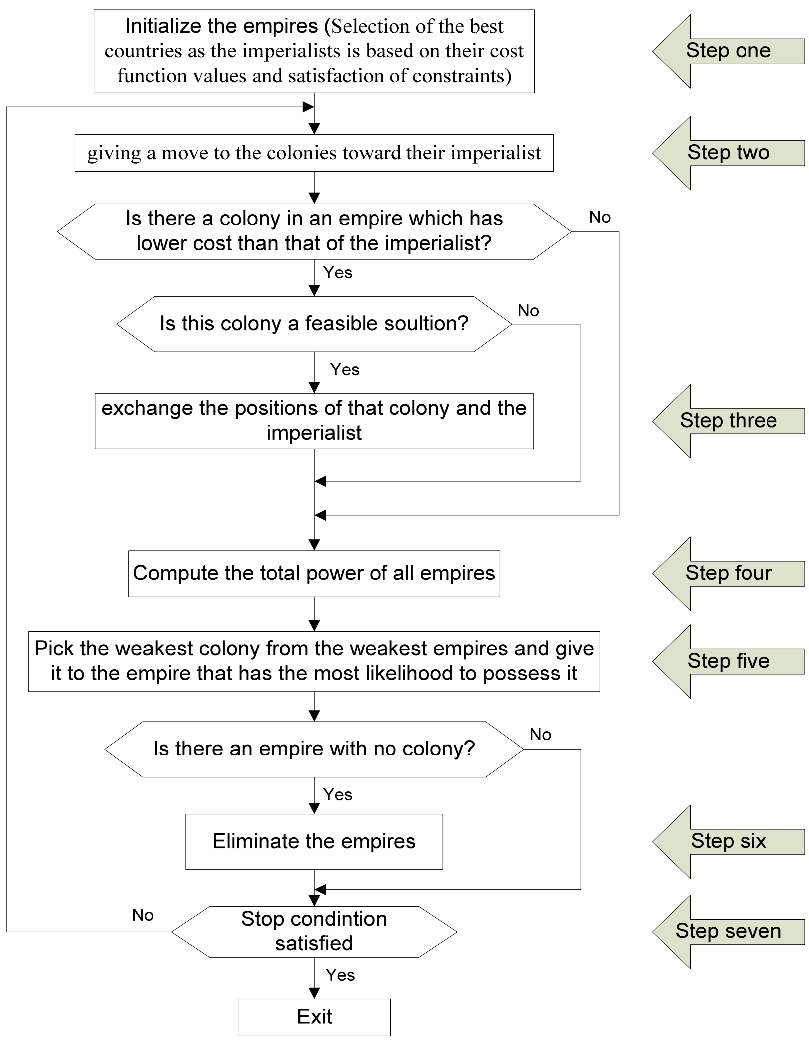

Figure 4- Flowchart of the ICA

3. Mathematical statement of optimizing discrete structural problems

A structural optimization problem with discrete variables, can be formulated as a nonlinear programming problem. In the category of sizing optimization of a truss structure, the cross-section areas of the members are considered as the design variables. Each of the design variables is chosen from a list of discrete cross-sections based on production standard. In that case, the objective function would be the structure weight. The design cross-sections must also satisfy some inequality constraints equations, which restrict the discrete variables. any structural optimization with discrete variables can be presented as follow [4]:

$$\begin{aligned} &\min && f(x^1, x^2, \ldots, x^d) \\ &\text{subject to}: && \\ & && g_q(x^1, x^2, \ldots, x^d) \le 0, \quad d = 1,2,\ldots, N_{\text{var}} \\ & && \qquad\qquad\qquad\qquad\quad q = 1,2,\ldots, M \\ & x^d \in S_d = \{X_1, X_2, \ldots, Xp\} && \end{aligned} \tag{16}$$

Function $f(x^1, x^2, \ldots, x^d)$ is the objective function which describe the weight of the truss, where $x^1, x^2, \ldots, x^d$ are a set of design variables. Sd consists of all permissive discrete variables $\{X_1, X_2, \ldots, Xp\}$ and x^d belongs to it. The inequality constraints are represented by $g_q(x^1, x^2, \ldots, x^d) \le 0$. The numbers of design variables and inequality constraints are shown by N_{var} and M, respectively. The number of available variables is represented by p [4].

4. Numerical Examples

Following examples show that the imperialist competitive algorithm, in optimizing the trusses, in comparison with algorithms such as heuristic particle swarm optimization (HPSO), particle swarm optimizer with passive congregation (PSOPC) and particle swarm optimization (PSO) gives better results and faster convergence.

This algorithm is coded in MATLAB and is run with a Pentium 4, 2GH computer. In all of examples, the population is equal to 50. In the equations (6) and (7), β=2 and $\gamma = \frac{\pi}{4}$ rad are assumed to have a good convergence of countries to the global minimum [9]. The number of the imperialist countries is considered as 4 and ξ=0.05. In all the following examples, the finite element method [FEM] is used for analysis.

4.1. 52-bar Planar Truss

In the Figure (5), a 52-bar planar truss is shown which has been analyzed by Wu [5] and Lee [14]. The material density and the modulus of elasticity are $7800 \, {}^{Kg}\!/\!{}_{m^3}$ and $E = 2.07 \times 10^5 \, Mpa$, respectively. The stress limitation for each member of this structure is equal to $\pm 180 Mpa$. In this example, there are 12 design variables. The members of this structure are divided into 12 groups: (1)A_1-A_4, (2) A_5-A_{10}, (3) A_{11}-A_{13}, (4) A_{14}-A_{17}, (5) A_{18}-A_{23}, (6) A_{24}-A_{26},(7) A_{27}-A_{30}, (8) A_{31}-A_{36}, (9) A_{37}-A_{39}, (10) A_{40}-A_{43}, (11) A_{44}-A_{49},and (12) A_{50}-A_{52}. The design variables are selected from Figure (6). The vertical loads used in this example are P_x=100KN, P_y=200KN. The maximum of iteration is considered as 3000 iterations.

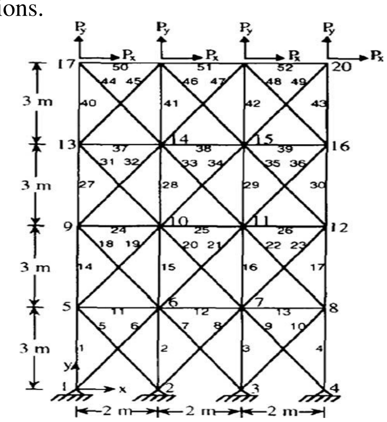

Figure 5- A 52-bar planar truss structure

No.	in.^{2}	mm^2	No.	in.^{2}	mm^2
1	0.111	71.613	33	3.840	2477.414
2	0.141	90.968	34	3.870	2496.769
3	0.196	126.451	35	3.880	2503.221
4	0.250	161.290	36	4.180	2696.769
5	0.307	198.064	37	4.220	2722.575
6	0.391	252.258	38	4.490	2896.768
7	0.442	285.161	39	4.590	2961.284
8	0.563	363.225	40	4.800	3096.768
9	0.602	388.386	41	4.970	3206.445
10	0.766	494.193	42	5.120	3303.219
11	0.785	506.451	43	5.740	3703.218
12	0.994	641.289	44	7.220	4658.055
13	1.000	645.160	45	7.970	5141.925
14	1.228	792.256	46	8.530	5503.215
15	1.266	816.773	47	9.300	5999.988
16	1.457	939.998	48	10.850	6999.986
17	1.563	1008.385	49	11.500	7419.340
18	1.620	1045.159	50	13.500	8709.660
19	1.800	1161.288	51	13.900	8967.724
20	1.990	1283.868	52	14.200	9161.272
21	2.130	1374.191	53	15.500	9999.980
22	2.380	1535.481	54	16.000	10322.560
23	2.620	1690.319	55	16.900	10903.204
24	2.630	1696.771	56	18.800	12129.008
25	2.880	1858.061	57	19.900	12838.684
26	2.930	1890.319	58	22.000	14193.520
27	3.090	1993.544	59	22.900	14774.164
28	1.130	729.031	60	24.500	15806.420
29	3.380	2180.641	61	26.500	17096.740
30	3.470	2238.705	62	28.000	18064.480
31	3.550	2290.318	63	30.000	19354.800
32	3.630	2341.931	64	33.500	21612.860

Figure 6- The available cross-section areas of the ASIC code [4]

In Table (1), the results obtained from other methods of optimizing of 52-bar planar truss have been compared with the results of ICA method. As it can be seen from the mentioned table, the imperialist competitive method has better results. Table (2) shows Statistical results. Figures (7), gives the comparison of convergence rates of 52-bar planar truss structure. It can be seen that the ICA method has better convergence rate. The ICA algorithm finds the best solution in 222 iterations (11100 analyses).

Table 1- The results of the 52-bar truss optimization

Area group	**ICA**	**Li et al [4]**			**Kaveh et al [10]**	**WU[5]**	**Lee[14]**
		HPSO	**PSO**	**PSOPC**	**HPSACO**	**GA**	**HS**
1	4658.055	4658.055	4658.055	5999.988	4658.055	4658.055	4658.055
2	1161.288	1161.288	1374.190	1008.380	1161.288	1161.288	1161.288
3	363.225	363.225	1858.060	2696.770	494.193	645.160	506.451
4	3303.219	3303.219	3206.440	3206.440	3303.219	3303.219	3303.219
5	940.000	940.000	1283.870	1161.290	1008.385	1045.159	940.000
6	494.193	494.193	252.260	729.030	285.161	494.193	494.193
7	2238.705	2238.705	3303.220	2238.710	2290.318	2477.414	2290.318
8	1008.385	1008.385	1045.160	1008.380	1008.385	1045.159	1008.385
9	641.289	388.386	126.450	494.190	388.386	285.161	2290.318
10	1283.868	1283.868	2341.93	1283.870	1283.868	1696.771	1535.481
11	1161.288	1161.288	1008.38	1161.290	1161.288	1045.159	1045.159
12	494.193	792.256	1045.16	494.190	506.451	641.289	506.451
Weight	**1903.366**	**1905.495**	**2230.16**	**2146.63**	**1904.83**	**1970.142**	**1906.76**

Table 2- Statistical results for 52-bar truss

	best	The results of the ICA based on 26 independent calculation								
ICA	1903.366	1903.366	1903.366	1903.366	1903.366	1903.366	1903.366	1903.366	1903.366	
		1903.366	1903.366	1903.366	1903.366	1903.366	1903.366	1903.366	1903.366	**worst**
		1903.366	1903.366	1903.700	1903.700	1912.646	1914.836	1915.283	2143.940	2203.558

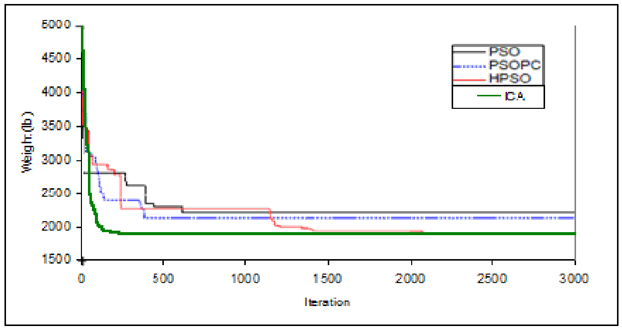

Figure 7- Comparison of convergence rates for the 52-bar planar truss structure

4.2. 72-bar Spatial Truss
A 72-bar spatial truss is shown in the Figure (8). This truss has been analyzed by Wu [5] and Lee [14].

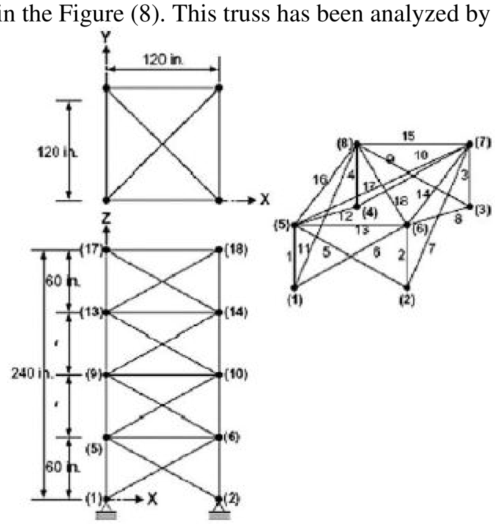

Figure 8- - A 72-bar spatial truss structure

The material density and the modulus of elasticity have been $0.1\, lb/in^3$ (0.0272 N/cm³) and $E = 10^4\, ksi$ (68947.57Mpa), respectively. The stress limitation for each member of this structure is equal to $\pm 25 ksi$ ($\pm$ 172.37Mpa) and the allowable displacement for each node in three directions is $\pm 0.25 in$ ($\pm$ 0.635 cm). This spatial truss is subjected to the two loading conditions shown in Table (3). In this example, cross-sectional areas (design variables) are divided to 16 groups: (1) A_1-A_4, (2) A_5-A_{12}, (3) A_{13}-A_{16}, (4) A_{17}-A_{18}, (5) A_{19}-A_{22}, (6) A_{23}-A_{30}, (7) A_{31}-A_{34}, (8) A_{35}-A_{36}, (9) A_{37}-A_{40}, (10) A_{41}-A_{48}, (11) A_{49}-A_{52}, (12) A_{53}-A_{54}, (13) A_{55}-A_{58}, (14) A_{59}-A_{66} (15), A_{67}-A_{70}, and (16) A_{71}-A_{72}. The design variables are selected from the set D=[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2](in^2). The maximum of iteration is considered as 1000 iterations.

Table 3- The loads applied to 72-bar truss

Node	**Load case 1**			**Load case 2**		
	F_x kips (kN)	**F_y kips (kN)**	**F_z kips (kN)**	**F_x kips (kN)**	**F_y kips (kN)**	**F_z kips (kN)**
17	5.0 (22.25)	5.0 (22.25)	-5.0 (22.25)	0.0	0.0	-5.0 (22.25)
18	0.0	0.0	0.0	0.0	0.0	-5.0 (22.25)
19	0.5	0.0	0.0	0.0	0.0	-5.0 (22.25)
20	0.5	0.0	0.0	0.0	0.0	-5.0 (22.25)

In Table (4), the results obtained from the ICA method have been compared with the results of other methods. As it can be seen the results of the imperialist competitive algorithm are better than the results of other methods. Table (5) shows Statistical results. In Figure (9), it can be seen that the ICA method has desirable convergence. The ICA algorithm finds the optimum in 203 iterations.

Table 4- The results of the 72-bar truss optimization

Area group	**ICA**	**Li et al [4]**			**Kaveh et al [10]**	**WU[5]**	**Lee[14]**
		HPSO	**PSO**	**PSOPC**	**HPSACO**	**GA**	**HS**
1	1.9	2.1	2.6	3.0	1.9	1.5	1.9
2	0.5	0.6	1.5	1.4	0.5	0.7	0.5
3	0.1	0.1	0.3	0.2	0.1	0.1	0.1
4	0.1	0.1	0.1	0.1	0.1	0.1	0.1
5	1.4	1.4	2.1	2.7	1.3	1.3	1.4
6	0.5	0.5	1.5	1.9	0.5	0.5	0.6
7	0.1	0.1	0.6	0.7	0.1	0.2	0.1
8	0.1	0.1	0.3	0.8	0.1	0.1	0.1
9	0.5	0.5	2.2	1.4	0.6	0.5	0.6
10	0.5	0.5	1.9	1.2	0.5	0.5	0.5
11	0.1	0.1	0.2	0.8	0.1	0.1	0.1
12	0.1	0.1	0.9	0.1	0.1	0.2	0.1
13	0.2	0.2	0.4	0.4	0.2	0.2	0.2
14	0.6	0.5	1.9	1.9	0.6	0.5	0.5
15	0.4	0.3	0.7	0.9	0.4	0.5	0.4
16	0.6	0.7	1.6	1.3	0.6	0.7	0.6
Weight	**385.54**	**388.94**	**1089.88**	**1069.79**	**385.54**	**400.66**	**387.94**

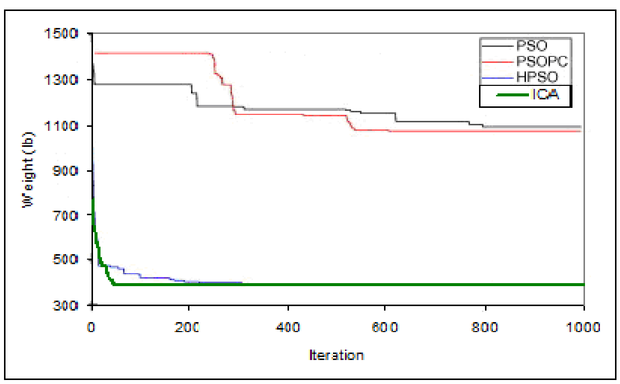

Figure 9- Comparison of convergence rates for the 72-bar spatial truss structure

Table 5- Statistical results for 72-bar truss

	best	The results of the ICA based on 26 independent calculation								
ICA	385.543	385.543	385.543	385.543	385.543	385.543	385.543	385.543	385.543	
		385.543	385.543	385.543	385.543	386.537	386.537	386.809	386.809	**worst**
		386.809	386.809	386.948	386.948	386.948	386.948	387.943	387.943	387.943

5. Conclusion

In this paper, an Imperialist Competitive Algorithm dealing with discrete variables is studied. The ICA with discrete variables has all advantages that it has with continuous variables. And as it is clear from the figures, this method has a convergence rate for discrete variables much better than the other methods such as PSO, HPSO and PSOPC.

In this paper, the Imperialist Competitive Algorithm has been tested for two truss structures and the calculated optimum solutions show that the ICA algorithm is as good as, and even in some cases better than the other methods regarding the accuracy and convergence rate. Also in this method different kind of constraints can be simply considered.

6. Refrences

[1] Kaveh, A., Talatahari, S.: Size optimization of space trusses using Big Bang–Big Crunch algorithm. Computers and Structures. 87, 1129–1140 (2009).

[2] Rahami, H., Kaveh, A., Gholipour, Y.: Sizing, geometry and topology optimization of trusses via force method and genetic algorithm. Engineering Structures. 30, 2360–2369 (2008).

[3] Rasmussen, M.H., Stolpe, M.: Global optimization of discrete truss topology design problems using a parallel cut-and-branch method. Computers and Structures. 86, 1527–1538 (2008).

[4] Li, L.J., Huang, Z.B., Liu, F.: A heuristic particle swarm optimization method for truss structures with discrete variables. Computers and Structures. 87, 435–443 (2009).

[5] Wu, S.J., Chow, P.t.: Steady-state genetic algorithms for discrete optimization of trusses. Comput Struct. 56, 979–991 (1995).

[6] Kirkpatrick, S., Gelatt, C., Vecchi, M.: Optimization by simulated annealing. Science 220(4598), 671–680 (1983).

[7] Perez, R.E., Behdinan, K.: Particle swarm approach for structural design optimization. Comput Struct. 85, 1579–1588 (2007).

[8] Atashpaz-Gargari, E., Lucas, C.: Imperialist Competitive Algorithm: An Algorithm for Optimization Inspires by Imperialistic Competition. IEEE Congress on Evolutionary Computation, Singapore. (2007).

[9] Khabbazi, A., Atashpaz-Gargari, E., Lucas, C.: Imperialist competitive algorithm for minimum bit error rate beamforming. Int. J. Bio-Inspired Computation. 125–133 (2009).

[10] Kaveh, A., Talatahari, S.: A particle swarm ant colony optimization for truss structures with discrete variables Journal of Constructional Steel Research. 65, 1558-1568 (2009).

[11] Rajeev, S., Krishnamoorthy, C.S.: Discrete optimization of structures using genetic algorithm. J Struct Eng, ASCE. 118(5), 1123–250 (1992).

[12] Ringertz, U.t.: On methods for discrete structural constraints. Eng Optim. 13(1), 47–64 (1988).

[13] Zhang, Y.n., Liu, J.p., Liu, B., Zhu, C.Y., Li, Y.: Application of improved hybrid genetic algorithm to optimize. J South China Univ Technol. 33(3), 69–72 (2003).

[14] Lee, K.s., Geem, Z.w., Lee, S.h., Bae, K.w.: The harmony search heuristic algorithm for discrete structural optimization. Eng Optim. 37(7), 663–84 (2005).