The Topology and Weight Optimization of a Truss Using Imperialist Competitive Algorithm (ICA)

Arash Mohammadzadeh Gonabadi 1,a, Mohsen Mohebbi 1,b, Ali Sohan Ajini 1,c

1 – Department of Mechanical Engineering, Islamic Azad University, Parand Branch, Tehran, Iran
a – arash_mg@semnan.ac.ir
b – Mohsen_mohebbi@ymail.com
c – alisohanajini@gmail.com

DOI 10.2412/mmse.33.83.364 provided by Seo4U.link

Keywords: optimization, truss, imperialist competition, applying constraints, topology optimization, weight optimization

ABSTRACT. The main point of structure optimization, although being economical, is that none of the engineering conditions and criteria must be neglected. Various methods such as Genetic Algorithm, Particle Swarm Algorithm, Annealing Algorithm and so on, have been used for optimizing the structures up to now. In this research, a new method called ‘The Imperialist Competitive Method (ICA)’, which is inspired by a social-human phenomenon, is used in topology, dimension and weight optimization of a structure and it is shown that is even useful for constraint problems as well. Conclusions show that the resulting diameters for structure partial sections are better than other optimization methods.

Introduction. From past decades, several optimization algorithms have been widely used in trusses. Therefore, the design of truss structures is one of the active research branches in optimization. Recently, some articles have used methods such as Genetic Algorithms [1], Particle Swarm Algorithm [2], and other random search methods (which are inspired by nature) in truss optimization. In this article, a new algorithm called ‘The Imperialist Competitive Method (ICA)’ is used. This algorithm was used by Esmaeel Atashpaz and his partners for the first time [3, 4] and it is inspired by a social-human phenomenon (not a natural one). Among the features of this algorithm we can point to the novelty of basic idea of algorithm, the high potential of optimization compared with other algorithms and appropriate convergence speed. The paper is organized in a way that ICA will be explained in the next section. In the third part, we formulate and apply constraints and compare the results of ICA with other optimization methods. The results show ICA is more convergent and functional than other methods.

In this research, ICA is used in topology, dimension and weight optimization of a structure and it is shown that's even useful for constraint problems as well. Conclusions show that the resulting diameters for structure partial sections are better than other optimization methods.

Imperialist competitive algorithm method

Imperialist competitive algorithm starts with an initial population like other methods. In this algorithm, each element of population is called a country. Countries are divided into two parts, Colony and Imperialist, and each imperialist, has colonized some of the colonies and controls them, depending on its power. Each imperialist country makes an empire with its colonies.

At first, we create an array using variables of the problem, which we are to be optimum. In this algorithm, this array is called a country. In an optimization problem, the next $N_{pAI}$ of a country is a $1 \times N_{pAI}$ array that is defined as follows:
We are going to find the best country by solving an optimization problem using this method. In fact, finding this country is equivalent to finding the best parameters for the problem which produce the lowest cost function in variables. The cost of a country can be found by evaluating the function in variables. (Equation 2)

\[
\text{Cost}_i = f(\text{country}_i) = [p_1, p_2, p_3, \ldots, p_{N_{\text{var}}}] \tag{2}
\]

To start algorithm, we create the initial country \( N_{\text{country}} \). Some of the best members of this population (the lowest cost function countries) are chosen as imperialist countries out of \( N_{\text{im}} \). The remained countries \( (N_{\text{col}}) \); create colonies which belong to an empire. For distributing the first colonies among empires, we give some of colonies to each empire depending on its power, thus the initial number of colonies for each empire is: [4] and [7-15]

\[
N.C_i = \text{round} \left\{ \frac{c_n}{\sum_{i=1}^{N_{\text{im}}}} \cdot (N_{\text{col}}) \right\}, i = 1, 2, \ldots, N_{\text{im}} \tag{3}
\]

In equation (3), \( N.C_i \), is the initial colonies of an empire and \( N_{\text{col}} \) is the total number of colonies existing among initial countries. Then we create \( N.C_i \) number of initial colonies randomly and give it to the \( n \)-th empire. The total power of an empire is defined as the sum of total imperialist countries plus a percentage of average power of colonies [4], as well. This is explained in equation (4).

\[
T.C_n = \text{cost(power of imperialist)} + \%\xi \{\text{cost(power of colonies)}\} \tag{4}
\]

In the above equation, \( T.C_n \) is the total power of \( n \)-th empire and \( \xi \) is a positive number which is between 0 and 1 and is assumed near 1. Assuming the lowest amount for \( \xi \) makes the total cost of an empire nearly equal to the cost of its central government (colony). Assimilation and Imperialist competition are two main pillars of this algorithm. Based on Assimilation, the imperialist countries try to trans shape other countries by cultural and customs change. This phenomenon can be seen in figure 1 [3] and [7-15].

\[\text{Imperialist}\]

\[\text{Colony}\]

\[\theta\]

\[x, d\]

**Fig. 1. The colony movement towards imperialist.**
In the colony movement towards imperialist $\theta$ and $\chi$ are random numbers with a uniform distribution and $\ell$ is the distance between colony and imperialist (Equations 5 and 6) [3].

$$\chi \sim U(0, \beta \times d), \beta > 1$$  \hspace{1cm} (5)

$$\theta \sim U(-\gamma, \gamma)$$  \hspace{1cm} (6)

$\beta$ is bigger than 1 and approaches 2, $\beta = 2$ can be an appropriate choice and $\gamma$ is an optional number. These show a limitation of which the colonies are moving near imperialist countries. If a colony achieves a better success than an imperialist does, they will be replaced.

Based on imperialist competition, each empire that cannot improve its power and loses challenging power will be eliminated. This takes place gradually meaning that the most powerful empire will seize these colonies over time and improves its power. Finally, there will be a competition among all empires to seize these colonies. To calculate the seize probability of the colony in the competition, at first, we must define the normalized total cost based on total cost of empire (Equation 7).

$$N.T.C_n = MAX\{T.C_i\} - TC_n$$  \hspace{1cm} (7)

In this equation, $T.C.n$ is the total cost of $n$-th empire and $N.T.C.n$ is the normalized total cost of that empire. By having normalized total cost, the probability of seizing the colony of competition by every empire will be achieved.

$$p_n = \frac{N.T.C.n}{\sum_{i=1}^{N_{emp}} N.T.C.i}, n = 1, 2, ..., N_{emp}$$  \hspace{1cm} (8)

We must consider that, mentioned colonies will be seized by the strongest empire; in fact, stronger empires have more probability to seize.

By having the probability of seize for each empire, a mechanism like Rolette cycle will be needed to give one of the empires the competition colony depending on the power of empires. In addition to the possibility of using Rolette cycle, a new mechanism is used which lower calculation has cost than Rolette cycle. This mechanism eliminates several operations for the calculation of the Cumulative Distribution Function (CDF) in Rolette cycle and only needs Probability Density Function (PDF) [4].

Finally, the imperialist competition makes a condition of which only one empire survives. This condition happens in ICA when it reaches an optimum point and algorithm stops. [4]

The steps of this algorithm are as follows:

1- Making the random nodes on selection function and make initial empires.
2- Moving colonies towards the empires. (Assimilation, Equation 5 and 6).
3- If there is a colony in an empire that reaches a better position than the imperialist, they will be replaced.
4- Calculating the total power of an empire. (Equation 4)
5- Choosing a colony from the weakest empire and give it to the empire, which has the most probability of, seize (Imperialist competition).
6- If the pause condition is satisfied (only one empire is remained, maximum number of repeats, time and so on), the algorithm will be stopped, otherwise we go back to step 2.

**Applying constraints and formulation**

To apply the constraints on target functions, two methods are suggested below:

The most general method that is usable for mentioned algorithm is the conversion of a constrained problem to an unconstrained one, using penalty function. We change target function and applied constraint method as follow [5] and [7-10]:

\[
(X, r_p) = F(X) + r_p P(X) \\
P(X) = \sum_{j=1}^{m} \{\max\{0, g_j(X)\}\}^2
\]

(9)

(10)

In the above equation, \(\phi(X, r_p)\) is the target function with applied penalty, \(F(X)\) target function, \(r_p\) a positive penalty parameter, \(g_j\) the applied constraints on the problem and \(m\) is the number of constraints.

In the second method, we consider these conditions:

First Condition. We consider an imperialist as a country that, in addition to having more power, satisfies the constraints applied to the problem.

Second Condition. If constraints are not applied due to movement deflection of colonies towards imperialist, deflection must be prevented and movement must be done directly to imperialist.

**Truss Optimization formulation**

In this section, the mathematical formulation methods of an optimization problem are explained.

According to this, cross section vector \([A]\) must be determined in a way that weight target function reaches the minimum amount (Equation 11).

\[
m = f(A) = \sum_{j=1}^{m} \rho_j L_j A_j
\]

We usually consider these constraints for optimization:

- \([G_1]\) = Basic Node
- \([G_2]\) = Kinematic Stability
- \([G_3]\) = \(\sigma_{th} - \sigma_j \geq 0 \quad j = 1, \ldots, m\)
- \([G_4]\) = \(\sigma_{max} - \sigma_k \geq 0 \quad k = 1, \ldots, n\)
- \([G_5]\) = \(A_i^{min} \leq A_i \leq A_i^{max} \quad i = 1, \ldots, m\)

In the above equation \(\rho\) – is density, \(L\) – is the length of every linkage, \(A\) – is the cross section of every link and \(m\) is the member of a trust. In the following section, we analyze these constraints.

For the beginning, we can use a ground structure. In a perfect ground structure (perfect graph), if \(n\) is the number of structure groups, we will have \(m = \binom{n}{2}\) links. It is not necessary for all \(m\) links to be in ground structure because a perfect ground structure, which has several links, increases the local optimization gain probability. Thus, we can choose a subset of \(m\) and start the process.
So, from now on, we call “m” the numbers of structure links

**G1 constraint**

In every truss structure, we have a series of important nodes that must be in every existing structure for a reasonable design. These important nodes are called basic nodes. These nodes are the ones that force is applied on or are on support.

![Basic Nodes](image)

Fig. 2. The basic nodes.

Number 1, 2 and 3 are the basic or ground nodes, must be in every design and we cannot eliminate them at all. So, we must check whether this constraint is satisfied or not, in each step. If it is not satisfied, this structure is not acceptable and cannot be chosen.

**G2 constraint**

This constraint checks the kinematic stability of structure. This means that the structure must not be a mechanism. At first, we use Garbler criterion for this constraint. For this purpose, the static degree of Indeterminacy of the structure is calculated using equation (12) before starting the structure analysis operation. Here is an example for Garbler criterion:

![Kinematic Stability Analysis](image)

Fig. 3. The analysis of kinematic stability constraint.

\[ D = m + r - 2j \]  \hspace{1cm} (12)

where \( m \) – the number of members (links) in every step of algorithm processing;

\( r \) – the number of support reactions;
If \( D < 0 \) the structure will be a mechanism and it is unstable thus unacceptable. Therefore, we do not do any more calculations for it.

This criterion is necessary but not enough for stability. Stiffness matrix of structure is used to analyze internal instability in optimization process. This means when we process problem for structure analysis, we achieve stiffness matrix \((K)\) and it's positive definite will be examined.

From a kinematic approach, a steady structure is the one that has symmetric stiffness matrix and positive definite matrix. If this condition does not satisfy, this structure is not acceptable and the calculations must be redone with another structure.

**G_3 constraint**

This constraint checks the amount of stress in all existing links meaning that after finite element analysis and finding the amount of stress for each member, the amount of stress in every member must be less than given allowable \((\sigma_{alt})\) stress in problem description [16-22].

**G_4 constraint**

After achieving the results of finite element analysis, the amount of achieved deflection in nodes must be less than allowable deflection.

**G_5 constraint**

In the second step of optimization (Dimension optimization) that we find the optimum cross section for each link, the cross section of every link must be in \( [A_{min} , A_{max}] \).

**Overlapping Constraints**

There must be no overlapping and duplication in a structure. For example, in figure 4, the link which has connected node 1 to node 3 is extra between 1, 2, and 3 nodes and must be eliminated. Such links must not be created in algorithm [16-22].

![Fig. 4. The overlapping links](image)

The second row is extra, so it is removed.

**Single-linked constraint**

If a node is used in topology matrix just once and is not a basic node, the link connected to that node must be eliminated.
The link between node 3 and node 4 is single and this must happen in nowhere in the algorithm. In this topology matrix, node 4 is only used once in this matrix thus the connected link to node 4 must be eliminated.

The constraint of Duplicated link

The structure is examined to eliminate the duplicated link in topology matrix [7-15].

The constraint of Empty link

The structure is examined to eliminate the empty link (which is only representing one node) in topology matrix. For example:
When all these initial constraints are checked, now we transfer the structure to finite element for analysis to find the stress amount in links, nodes deflection and check whether the stress and deflection constraints have contravention or not. When stress and deflection constraints are satisfied, the ICA will be applied, a new topology will be achieved and the algorithm is repeated thus the target function which is the weight of the structure will be achieved and the minimum weight of the structure will be announced as the optimum structure.

**Topology optimization process Steps**

1. Selection of an initial topology
2. Analysis of initial constraints (If satisfied go to the next step, if not so, go back to the first step)
3. Structure analysis and finding the amount of stress and deflection
4. Checking whether steadiness, stress and movement constraints are satisfied (if not so, go back to the first step, if so, go to the nest step)
5. Applying ICA and achieving a new topology
6. Repeating this process until reaching stoppage condition

**Size Optimization**

As we have a two-step method, in the second step when the optimum topology is achieved, we begin to process size optimization. In this step, it is perfectly clear whether a link exists or not and only cross section of existing links in structure is optimum. The link cross sections are chosen continuously [7-15]. Therefore, we use ICA which is naturally a continuous algorithm. The target function is the weight of the structure, which is the result of this equation:

\[
m = f(A) = \sum_{j=1}^{m} p_j L_j A_j
\]

When all these initial constraints are checked, now we transfer the structure to finite element for analysis to find the stress amount in links, nodes deflection and check whether the stress and deflection constraints have contravention or not. When stress and deflection constraints are satisfied, the ICA will be applied, a new topology will be achieved and the algorithm is repeated thus the target function which is the weight of the structure will be achieved and the minimum weight of the structure will be announced as the optimum structure.

**Topology optimization process Steps**

1. Selection of an initial topology
2. Analysis of initial constraints (If satisfied go to the next step, if not so, go back to the first step)
3. Structure analysis and finding the amount of stress and deflection
4. Checking whether steadiness, stress and movement constraints are satisfied (if not so, go back to the first step, if so, go to the nest step)
5. Applying ICA and achieving a new topology
6. Repeating this process until reaching stoppage condition

**Size Optimization**

As we have a two-step method, in the second step when the optimum topology is achieved, we begin to process size optimization. In this step, it is perfectly clear whether a link exists or not and only cross section of existing links in structure is optimum. The link cross sections are chosen continuously [7-15]. Therefore, we use ICA which is naturally a continuous algorithm. The target function is the weight of the structure, which is the result of this equation:

\[
m = f(A) = \sum_{j=1}^{m} p_j L_j A_j
\]
There is a limitation for members’ cross section meaning that all members’ cross sections must be in this limit:

\[ A_{\text{min}} \leq A_j \leq A_{\text{max}} \]  

(18)

Hence in size optimization, design parameters are members' cross sections which are chosen from \([A_{\text{min}}, A_{\text{max}}]\) continuously. We must consider components for each particle as long as design parameters (number of members in second step). Meaning that a particle is a vector that has components as long as design parameters [16-22].

\[ X_i = [A_{i1}, A_{i2}, ..., A_{in}] \]  

(19)

where \(n\) – the number of design parameters.

**Numerical Example**

It is shown in the below example that ICA has more functionality than Genetic and POS algorithm. This algorithm is coded in Matlab [23, 25]. In this algorithm, in the first step of optimization (Topology optimization), the number of population is 10 and the number of imperialist countries are 3 and in equation (5), \(\beta\) is 2 and in equation (4) \(\xi\) is 0.05 [16-22].

After achieving optimum type, we go into the next step which is the optimization for structure weight and diameters and in second step of optimization (size optimization), the number of population is 100 and the number of imperialist countries are 50 and in equation (5), \(\beta\) is 2 and in equation (4) \(\xi\) is 0.05.
In this example, there is a 10-linkage structure with 6 nodes. The position of nodes and boundary conditions are shown in figure and table (1) which is below.

Elasticity module: $10^7$ Psi=68965.5 MPa, Density: 0.1 lb/in$^3$=2768.096 kg/m$^3$, Maximum applied stress: 25000 Psi=172.41 Mpa, allowable displacement for every node in X and Y direction: 2 in=5.08 cm, Force (P)= 10000lb=444.8 KN , Linkage length (L) = 360 in=9.144 m , the maximum and minimum cross section $[A_{\text{min}}, A_{\text{max}}]$ = [0.01,35] in$^2$=[0.6452 225.806] [16-22].

After running the program, the structure is topologically optimum as you can see in figure 8 and its structure is different and has found its optimum states based on constraints and properties. We can see the convergence rate of the algorithm for this structure in figure 9.

In table 1 and table 2 we can see the results of optimization using ICA, Genetic algorithm and POS and the weight convergence rate can be shown in figure 10.
Table 1. Achieved cross sections of optimization.

<table>
<thead>
<tr>
<th></th>
<th>GA</th>
<th>POS [6]</th>
<th>ICA</th>
<th>Link cross section</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>27.9012 in$^2$</td>
<td>19.961 in$^2$</td>
<td>19.1776 in$^2$</td>
<td>Link cross section between node 1 &amp; 5</td>
</tr>
<tr>
<td></td>
<td>20.4780 in$^2$</td>
<td>25.0146 in$^2$</td>
<td>22.1729 in$^2$</td>
<td>Link cross section between node 1 &amp; 6</td>
</tr>
<tr>
<td></td>
<td>16/6802 in$^2$</td>
<td>22.7334 in$^2$</td>
<td>20.8352 in$^2$</td>
<td>Link cross section between node 3 &amp; 5</td>
</tr>
<tr>
<td></td>
<td>24/5390 in$^2$</td>
<td>15.0218 in$^2$</td>
<td>1.8352 in$^2$</td>
<td>Link cross section between node 3 &amp; 6</td>
</tr>
<tr>
<td></td>
<td>30/8563 in$^2$</td>
<td>27.9882 in$^2$</td>
<td>32.8931 in$^2$</td>
<td>Link cross section between node 4 &amp; 5</td>
</tr>
<tr>
<td></td>
<td>6.6060 in$^2$</td>
<td>5.947 in$^2$</td>
<td>6.0080 in$^2$</td>
<td>Link cross section between node 4 &amp; 6</td>
</tr>
</tbody>
</table>

Table 2. Achieved weights of optimization.

<table>
<thead>
<tr>
<th></th>
<th>GA</th>
<th>POS [6]</th>
<th>ICA</th>
<th>Link cross section</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10004.4432</td>
<td>718.596</td>
<td>690.3936</td>
<td>Link cross section between node 1 &amp; 5</td>
</tr>
<tr>
<td></td>
<td>737.208</td>
<td>900.5256</td>
<td>798.2244</td>
<td>Link cross section between node 1 &amp; 6</td>
</tr>
<tr>
<td></td>
<td>240.4872</td>
<td>818.4024</td>
<td>721.7568</td>
<td>Link cross section between node 3 &amp; 5</td>
</tr>
<tr>
<td></td>
<td>883.404</td>
<td>240.7848</td>
<td>66.0672</td>
<td>Link cross section between node 3 &amp; 6</td>
</tr>
<tr>
<td></td>
<td>1110.8268</td>
<td>1007.5752</td>
<td>1184.1516</td>
<td>Link cross section between node 4 &amp; 5</td>
</tr>
<tr>
<td></td>
<td>237.816</td>
<td>214.092</td>
<td>216.288</td>
<td>Link cross section between node 4 &amp; 6</td>
</tr>
<tr>
<td></td>
<td>4214.1852 lb</td>
<td>3899.976 lb</td>
<td>3677.4864 lb</td>
<td>Total weights of the structure</td>
</tr>
</tbody>
</table>

Summary. The given optimization algorithm can be used as a simple, quick, and appropriate optimization way for solving most optimization problems. This method will be converged to an appropriate solution during a timeline. On the other hand, we can simply apply several kinds of constraints in this optimization method.

In this article, ICA is used to optimize the truss. All results and figures are achieved due to the high potential of this algorithm in quick convergence and finding optimum solution.

References


