Optimum Design of Truss Structures Based on Differential Evolution Strategy

Zhaoliang Wang, Hesheng Tang, Pengfei Li
Research Institute of Structural Engineering and Disaster Reduction, Tongji University
Shanghai , China
smith333@163.com, thstj@mail.tongji.edu.cn, shangyiiceberg@163.com

Abstract—**A novel optimization method based on a Differential Evolution (DE) strategy for designing low-weight truss structures is presented for both continuous and discrete variables. Applications of this technique on the optimization of a benchmark-type truss structure with continuous and discrete variables are given to evaluate its effectiveness. Results are compared with various classical and evolutionary optimization methods which show that the proposed procedure based on DE outperforms other methods and can effectively be applied to the optimization problems of truss structures with both continuous and discrete design variables.**

Keywords-Optimum Design; Differential Evolution; Truss Structures

I. INTRODUCTION

In general, the optimization design is a non-convex and multi-peak problem. Thus, some traditional techniques, such as optimality criteria (OC) and mathematical programming (MP) methods will meet great challenges when handling such problem. In recent years heuristic computational intelligence methods belonging to the global optimization category have proven to be promising tools to solve the complex optimization problems of skeletal structures with both continuous and discrete variables. These have been found to be powerful methods in domains where traditional methods have not been proved to be effective. Among the most important heuristic optimization methods, such as genetic algorithm (GA), big bang-big crunch (BB-BC) optimization, ant colony optimization (ACO) and particle swarm optimization (PSO) have been successfully applied to a variety of optimization problems. Goldberg et al. used GA to optimize skeletal structure with continuous variables[1]. Rajeev et al. optimized structures with discrete variables based on GA[2]. Li et al. applied PSO in steel structures with continuous variables[3] and later in truss structures with discrete variables[4]. Camp designed a procedure based on the BB-BC optimization for both discrete and continuous variable optimization [5].

Differential Evolution (DE) is one of the recent- developed evolutionary optimization algorithm invented in 1995 for global optimization over continuous spaces[6]. Recently, the DE method has also been applied to the discrete optimization[7]. A comparative study of DE and other algorithms was made showing that DE algorithm outperforms very significantly other methods such as GA and PSO[8]. In the following years much attention were paid and efforts were made to improve the performance of DE as well as explore its application areas. The application fields of DE were enlarged. It's been successfully used in such fields as engineering design[9], reliability analysis[10] and system identification[11].

In this study, a DE-based method of low-weight design of the truss structure is developed. In addition, the continuous and discrete design variable models are investigated.

II. PROBLEM FORMULATION

In the optimization of a structure, the objective of a truss design is to minimize the total cost while satisfying design constraints. Typically, an optimal truss design is one whose total weight is minimized while not exceeding allowable values for compressive and/or tensile stress in each member and deflection of any connection. In these terms, a truss optimization problem can be described as to obtain the values for design variables $X=[X_1,X_2,\ldots,X_D]^T$, minimizing an objective function f $(X_1,X_2,\ldots,X_D)$, satisfying at the same time, the design constraints $g_i(X_1,X_2,\ldots,X_D)\leqslant 0 \quad i=1,\ldots,m$.

A. Design Variables

When the topology of a truss is fixed, the cost is directly related to the cross-sectional area of each member of the truss. Therefore, the discrete and/or continuous design variables are chosen as: $X=[X_1,X_2,\ldots,X_D]^T$,where X_i=the value of cross-sectional area of i-th member group. For continuous variables, $X_i\in[A^l, A^u]$, while for discrete variables, $X_i\in\{A^1, A^2,\ldots, A^N\}$.

B. Design Constraints

Design constraints vary depending on the situation or the level of analysis, but typically truss designs are limited by allowable material stresses, structural displacements and cross-section range: $\sigma_i^l\le\sigma_i\le\sigma_i^u;\delta_c^l\le\delta_c\le\delta_c^u;\ A_i^l\le X_i\le A_i^u$. Where σ_i is the stress of i-th member group bounded by an upper and lower limit: δ_c is the deflection of connection c bounded by an upper and lower limit; X_i is the cross-section area of i-th member group bounded by an upper and lower limit.

C. Objective Function

The objective function is used to evaluate the design of structures, which could for instance be the weight, the cost or any other relevant objectives. Here we choose the total weight as the objective function. For each truss design candidate,

Sponsor: National Natural Science Foundation of China (Project NO. 50708076)

stresses, deflections constraints are evaluated to determine whether the design is feasible. If a design is infeasible, a penalty function is applied to the structural weight reflecting the degree of constraint violation. The penalized weight helps depart from the infeasible designs and focus on feasible ones. As for cross-sectional constraints, we handle it in a different manner. Once a candidate solution violates the area bounded limits, it will be reset randomly within the search space. Thus, the formulation of the objective function is: $f = \sum_{i=1}^{n} \rho_i X_i L_i + \lambda M$, Where f =objective function; ρ_i, X_i, L_i =material density, cross-sectional area and length of i-th member respectively; M is a very large number(10E+30); λ =factor of penalty function, λ =0, if all constraints are satisfied; otherwise, λ =1.

III. DE ALGORITHM

The DE algorithm is a population-based algorithm similar to genetic algorithms using such operators as crossover, mutation and selection. In DE, a population of NP (population size) solution vectors is initialized randomly at the start, which is evolved to find optimal solutions through the repeated procedures of mutation, crossover, and selecting operation. In its canonical form, the DE algorithm is only capable of handling continuous variables. Later Lampinen and Zelinka (1999) discussed the way to modify DE for mixed variable and discrete optimization, in which only a couple of simple modifications are required[7].

Here we first describe the DE algorithm for continuous optimization, and in the end of this section we will present a few modifications to suit DE to discrete optimization.

An optimization task consisting of D parameters can be represented by a D-dimensional vector. Let $S \in R^D$ be the search space of the problem under consideration. Then, the DE algorithm utilizes NP, D-dimensional vectors $\boldsymbol{x_i}=(x_{i1},x_{i2},\ldots,x_{iD})^T \in S$, i=1,2,3…,NP as a population for each iteration, called a generation of the algorithm.

A. Mutation

The objective of mutation is to ensure search diversity in the parameter space as well as to direct the existing object vectors with suitable amount of parameter variation in a way which will lead to better results at a suitable time. It keeps the search robust and explores new areas in the search domain.

According to the mutation operator, for each individual, $\boldsymbol{x}_i^{(G)}$, i=1 ⋯ NP, at generation G, a mutation vector $\boldsymbol{v_i}^{(G+1)}=(v_{i1}^{(G+1)}, v_{i2}^{(G+1)},\ldots, v_{iD}^{(G+1)})^T$ is determined by the following equation[6]:

$$v_i^{(G+1)} = x_i^{(G)} + F_1(x_{best}^{(G)} - x_i^{(G)}) + F(x_{r1}^{(G)} - x_{r2}^{(G)}) \quad (1)$$

where $\boldsymbol{x_{best}}^{(G)}$ = best individual of the population at generation G; F and F1 > 0 = real parameters, called mutation constants, which control the amplification of difference between two individuals so as to avoid search stagnation; and r_1, r_2 are mutually different integers, randomly selected from the set {1, 2, ..., i -1, i +1, ..., NP}

B. Crossover

In DE the crossover operator is applied on the population after the mutation phase. For each mutant vector, $\boldsymbol{v_i}^{(G+1)}$, a trial vector $\boldsymbol{u_{ij}}^{(G+1)}=(u_{i1}^{(G+1)}, u_{i2}^{(G+1)},\ldots, u_{iD}^{(G+1)})^T$ is generated, with

$$u_{ij}^{(G+1)} = \begin{cases} v_{ij}^{(G+1)} & \text{if } (rand(j) \le CR) \text{ or } (j = randn(i)) \\ x_{ij}^{(G+1)} & \text{if } (rand(j) > CR) \text{ or } (j \ne randn(i)) \end{cases} \quad (2)$$

where j =1,2, ..., D; *rand(j)* is the j-th independent random number uniformly distributed in the range of [0, 1]. *randn*(*i*) is a randomly chosen index from the set {1, 2, ..., D}, and CR is user defined crossover constant ∈ [0, 1] that controls the diversity of the population.

C. Selection

After producing the offspring（trial vector）, the performance of each offspring vector $\boldsymbol{u}_i^{(G+1)}$ and its parent $\boldsymbol{x}_i^{(G)}$is compared. DE employs a greedy selection process that the better one of new offspring and its parent wins the competition and is retained in the population and passed to the next generation. Thus, if f denotes the objective function under consideration, then

$$x_i^{(G+1)} = \begin{cases} u_i^{(G+1)} & \text{if } f(u_i^{(G+1)}) < f(x_i^{(G)}) \\ x_i^{(G)} & \text{if } \quad otherwise \end{cases}, \quad (3)$$

These A,B,C steps are repeated until specified termination criterion is reached.

D. Operational parameter

DE has three key parameters: scaling factor of the difference vector – F, crossover control parameter – CR and population size – NP. An additional control variable, F1, is introduced to provide a means to enhance the greediness of the scheme by incorporating the current best vector $\boldsymbol{x_{best}}^{(G)}$. The operational parameters control the balance between exploitation and exploration, so as to increase the convergence velocity and robustness of the search process. Depending on the problem and available computational resources, the population size can be in the range from 2D (D is the problem dimension) to 100D [12]. In our experiments, with a population size of 20D, F1 = 0.95 and F = 0.8 appear to be reasonably good value to generate satisfactory results. The test results [6] show that a satisfactory range of CR appears to be within 0.8–1.0. In this study, we set DE parameters as：NP=50, F1=0.95, F= 0.8, CR=0.85, Max_it =300 and 250 respectively.

E. Modifications made to suit for discrete variables

For discrete optimization, the design variables are chosen from a list of discrete cross-sections, i.e. $X_i \in S=\{A^1, A^2,\ldots, A^N\}$. The optimization problem with discrete variables is a combination optimization problem which obtains its best solution from all possible variable combinations.

According to Lampinen and Zelinka[7], the basic idea for handling discrete problems is: Instead of optimizing the value of the discrete variable directly, optimize the value of its index *i*, only during evaluation is the indicated discrete value used. The details are as follows: First, number the discrete variables.

The scalar S includes all permissive discrete variables arranged in ascending sequence. Each element of the scalar S is given a sequence number to represent the value of the discrete variable correspondingly. It can be expressed as: S={A^1, A^2,...A^j..., A^N}, 1≤j≤N. where N is the number of all permissive discrete variables. Second, a mapped function h(j) is selected to index the sequence number of the elements in set S and represents the value of A^j of discrete variables correspondingly: h(j)= A^j. Thus, the sequence numbers of the elements will substitute for the discrete values in the scalar S. This method is used to search the optimum solution, and makes the variables to be searched in a continuous space.

The DE algorithm starts with NP solution vectors initialized randomly in the search space. The position of the i-th solution in the space can be described by a vector $\boldsymbol{x}_i$,

$$\boldsymbol{x}_i=(x_{i1},x_{i2},\ldots,x_{id},\ldots,x_{iD})^T,\quad 1\le d\le D,\ i=1,2,3\ldots,NP$$

The scalar $x_{id}\in$ {1, 2...j..., N} corresponds to the discrete variable set {A^1, A^2,...A^j..., A^N} by the mapped function h(j). Therefore, the solution vectors updates through the continuous space, but only stays at the integer space. In other words, all the components of the vector $\boldsymbol{x}_i$ are integer numbers. The solution is updated by Eqs (1), (2) and (4).

$$\boldsymbol{x}_i^{(G+1)}=\begin{cases}\boldsymbol{INT}(\boldsymbol{u}_i^{(G+1)}) & \text{if } (f(INT(\boldsymbol{u}_i^{(G+1)}))<f(INT(\boldsymbol{x}_i^{(G)})))\\ \boldsymbol{INT}(\boldsymbol{x}_i^{(G)}) & \text{if } (f(INT(\boldsymbol{u}_i^{(G+1)}))\ge f(INT(\boldsymbol{x}_i^{(G)})))\end{cases}\tag{4}$$

IV. Ten-Bar Truss Design Examples

The ten-bar truss has been widely employed by researchers and has effectively become a benchmark problem in the field of structural optimization. The ten-bar truss has been designed by many researchers using various approaches and techniques, such as the gradient search technique[13], OC[14] , GAs[15,16], PSO[17,18], ACO [19] and BB-BC [5]. This paper utilized the DE method to solve the low-weight designs of the general 10-bar truss with continuous and discrete design variable models.

The configuration of ten-bar cantilevered truss is shown in Fig. 2. The material has a modulus of elasticity of 10^7 psi and a mass density of 0.1 lb/in^3. The maximum allowable stress in any member of the truss is ±25 ksi; and the maximum deflection of any node (in both the vertical and horizontal direction) is ±2.0 in. In this study, the program is coded with MATLAB. For the convenience of comparison with results from other literatures, the English unit is adopted.

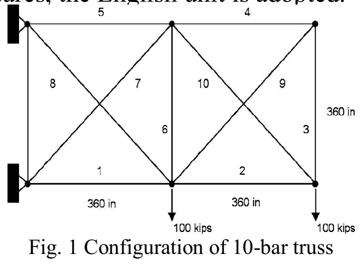

Fig. 1 Configuration of 10-bar truss

A. *Ten-Bar Truss Design Using Continuous Variables*

For the first ten-bar truss design example, the design variables are continuous. Cross-sectional areas may vary from 0.1 in^2 to 35.0 in^2. Formulate the optimization problem in a standard form as below:

$$\begin{cases} X=[X_1,X_2,X_3......X_{10}]^T\\ \min\quad F=\sum_{i=1}^{10}\rho_i X_i L_i+\lambda M\\ s.t\ \begin{cases}|\sigma_i|\le 25ksi & (i=1,2,\cdots,10)\\ |\delta_c|\le 2.0in & (c=1,2,\cdots,6)\\ 0.1in^2\le X_i\le 35.0in^2\end{cases}\\ \lambda=0\quad or\quad 1\end{cases}\tag{5}$$

To evaluate the effectiveness of DE algorithm in the optimization of truss structure, run this optimization program randomly for 5 times. The typical convergence history of objective function is drawn in Fig.2. The best and worst results are listed to be compared with results listed in other existing literatures (see Table 1).

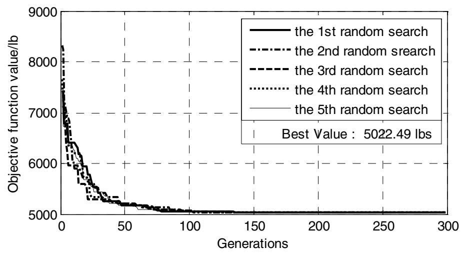

Fig 3. Convergence history of 10-bar truss using continuous variables

Table1. Design for 10-bar truss using continuous variables

Cross-sectional areas (in^2)					
Members	DE best	DE worst	GA[15]	PSO[17]	BB-BC[5]
1	22.261	22.680	24.07	23.268	22.344
2	15.305	15.423	13.96	15.129	15.437
3	0.938	0.899	0.56	0.554	0.967
4	0.100	0.101	0.10	0.100	0.100
5	31.070	30.871	28.92	29.999	30.804
6	0.100	0.100	0.10	0.100	0.100
7	21.854	21.792	21.95	21.232	21.890
8	5.811	5.795	7.69	7.454	5.801
9	0.100	0.101	0.10	0.100	0.100
10	21.548	21.421	22.09	21.670	21.532
Weight(lb)	5,022.486	5,022.866	5,076.31	5,059.85	5,022.65
W_{avg}(lb)	5022.658		5,067.51	5,041.92	5,025.97
W_{stdev}(lb)	0.163		--	17.509	2.746

Note: 1 in^2=6.452 cm^2; 1 lb=4.45 N.

Table 1 lists the best and worst design developed by the DE algorithm. The best weight value obtained of feasible truss in 300 runs is 5,022.486 lbs, the worst value is 5,022.866 lbs. The average weight 5022.658 lbs developed by DE is at least 0.89%, 0.38%, 0.07% lighter than the results developed by GA, PSO and BB-BC respectively. In addition, the standard deviation of the results based on DE is 0.163, which is much smaller than that of other literature listed. The results of Table 1 indicate that the proposed design procedure based on DE strategy exhibits a significant improvement in computational efficiency and robustness over those using GA, PSO and BB-BC.

B. *Ten-Bar Truss Design Using Discrete Variable*

In the second ten-bar truss design example, the following list of 41 discrete cross-sectional areas are available for each member（1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.88, 2.93, 3.09, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.5, 13.5, 13.9, 14.2, 15.5, 16.0, 16.9, 18.8, 19.9, 22.0, 22.9, 26.5, 30.0, 33.5 in^2）.

For such combination optimization problem with 41 discrete variables for every one of the 10 bars, the approximate size of the resulting search space is 1.34×10^{16} designs. 5 random searches are performed in this example. DE strategy is proved to be excellent in handling such problems.

The convergence history for design of the ten-bar truss is shown in Fig. 3. Table 2 lists the best design developed by the DE algorithm, a truss weighing 5,490.74 lbs, which is identical to that found by GA, ACO and BB-BC listed.

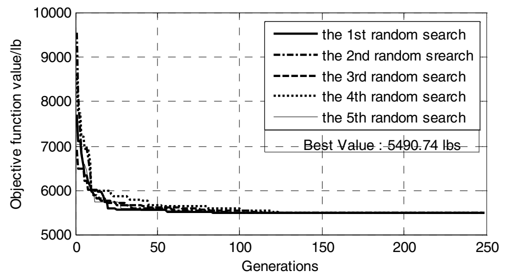

Fig 3. Convergence history of 10-bar truss using discrete variables

Table2. Designs for Ten-Bar Truss Using Discrete Variables

	Cross-sectional areas (in^2)				
Members	DE best	DE worst	GA[16]	ACO[19]	BB-BC[5]
1	22.9	22.9	22.9	22.9	22.9
2	14.2	14.2	14.2	14.2	14.2
3	1.62	1.62	1.62	1.62	1.62
4	1.62	1.62	1.62	1.62	1.62
5	33.5	33.5	33.5	33.5	33.5
6	1.62	1.62	1.62	1.62	1.62
7	22.9	22.9	22.9	22.9	22.9
8	7.97	7.97	7.97	7.97	7.97
9	1.62	1.62	1.62	1.62	1.62
10	22.0	22.0	22.0	22.0	22.0
Weight(lb)	5,490. 74	5,490. 74	5,490.74	5,490.74	5,490.74
W_{avg}(lb)	5,490. 74		--	5,510.52	5,494.17
W_{stdev}(lb)	0		--	23.19	12.420

In the 5 runs of the DE algorithm, the average weight of the best feasible truss from every run is 5,490.74 lbs with a standard deviation of 0 lbs, which means every search got the same solution. These values show a significant improvement of the DE approach as compared to GA, ACO and BB-BC listed in the table 2.

V. CONCLUSIONS

This paper has presented a novel differential evolution (DE) strategy for the problem of low-weight design of truss structures with both continuous and discrete variables. DE has the advantage of incorporating a relatively simple and efficient form of self-adapting mutation, crossover selection operation. Through a benchmark-type truss optimization problem, the DE algorithm demonstrated that it can routinely minimize the overall weight of truss structures while satisfying material and performance constraints. The results from our study based on DE is clearly and consistently superior compared to those based on GA, PSO, BB-BC for continuous optimization problems. For discrete design variables, the results are same but with a standard deviation of 0. All the above demonstrate the consistency and computational efficiency of the application of a DE strategy in the optimization problems of truss structures.

REFERENCES

[1] Goldberg, D. E., and Samtani, M. P. "Engineering optimization via genetic algorithm." *Proc.*, *Electronic Computation*, ASCE,1986, New York, 471–482

[2] Rajeev, S., and Krishnamoorthy, C. S. "Discrete optimization of structures using genetic algorithms." *J. Struct. Eng.*,1992,118(5), 1233–1250.

[3] Li LJ, Liu F, Ren FM, Wu QH. An improved particle swarm optimization method and its application in steel structures. International symposium on new Olympic, new shell and spatial structures, Beijing; 2006. p. 282–3.

[4] Li LJ, Huang ZB, Liu F. A heuristic particle swarm optimization method for truss structures with discrete variables. Comput Struct, 2009, 87:435-443.

[5] Camp, CV. Design of space trusses using big bang-big crunch optimization Journal of Structural Engineering-ASCE,2007,Vol.133(7): 999-1008.

[6] Storn R, Price,K. Differential evolution-A simple and efficient adaptive scheme for global optimization over continuous spaces [J]. Journal of Global Optimization,1997,11(4):341~359.

[7] Lampinen, J., Zelinka, I., 1999. Mechanical engineering design optimization by differential evolution. In: Corne, D., Dorigo, M., Glover (Eds.), New Ideas in Optimization. McGraw Hill, International (UK), pp. 127–146.

[8] Vesterstrom J., Thomsen R. A Comparative Study of Differential Evolution, Particle Swarm Optimization, and Evolutionary Algorithms on Numerical Benchmark Problems [J]. Congress on Evolutionary Computation, 2004, Vol.2:1980~1987

[9] Gong W., Cai Z., Zhu L. An efficient multi-objective differential evolution algorithm for engineering design [J]. Structural and Multidisciplinary Optimization,2009,4(2):137~157

[10] Coelho, Leandro dos Santos. Reliability-redundancy optimization by means of a chaotic differential evolution approach [J]. Chaos Solitons Fractals, 2009,7, Vol. 41(2), 594~602

[11] Tang, H., Xue, S., Fan C. Differential Evolution strategy for structural system identification [J]. *Comput. Struct.*.86(2008), 2004-2012

[12] Price K. An introduction to differential evolution. In: Corne D, Dorigo M, Glover F, editors. New ideas in optimization. London: McGraw-Hill; 1999. p. 79–108.

[13] Venkayya, V. B. "Design of optimum structures." *Int. J. Comput. Struct.*, 1971,1, 265–309.

[14] Allwood, R. J., and Chung, Y. S. "Minimum-weight design of trusses by an optimality criteria method." *Int. J. Numer. Methods Eng.*,1984, 20, 697–713.

[15] Camp, C., Pezeshk, S., and Cao, G. "Optimized design of two-dimensional structures using a genetic algorithm." *J. Struct. Eng.*,1998, Vol.124(5), 551–559.

[16] Mahfouz, S. Y. "Design optimization of structural steelwork." Ph.D. thesis, Dept. of Civil and Environmental Engineering,1999, Univ. of Bradford, United Kingdom.

[17] Fourie, P. C., and Groenwold, A. A. "The particle swarm optimization algorithm in size and shape optimization." *Struct. Multidiscip. Optim.*,2002, 23, 259–267.

[18] Schutte, J. J., and Groenwold, A. A. “Sizing design of truss structures using particle swarms.” *Struct. Multidiscip. Optim.*,2003, 25, 261–269.

[19] Camp, CV., and Bichon, B. J. “Design of space trusses using ant colony optimization.” *J. Struct. Eng.*,2004, Vol.130(5), 741–7