17

Hunting Search Optimization Modification and Application

Ferhat Erdal
Department of Civil Engineering
Akdeniz University, Turkey

Osman Tunca
Department of Civil Engineering
Karamanoglu Mehmetbey University, Turkey

Erkan Dogan
Department of Civil Engineering
Celal Bayar University, Turkey

CONTENTS

17.1 Introduction

Hunting search algorithm (HuS) is one of the recent additions to the meta-heuristic search techniques of combinatorial optimization problems, introduced by Oftadeh et al. [1]. This approach is based on the group hunt-

ing of animals such as lions, wolves and dolphins. The commonality in the way these animals hunt is that they all hunt in a group. They encircle the prey and gradually tighten the ring of siege until they catch the prey. Each member of the group corrects its position based on its own position and the position of other members during this action. If a prey escapes from the ring, hunters reorganize the group to siege the prey again. The hunting search algorithm is based on the way wolves hunt. The procedure involves a number of hunters which represents the hunting group which are initialized randomly in the search space of an objective function. Each hunter represents a candidate solution of the optimum design problem. This chapter is organized as follows. In Section 17.2 a brief introduction of the simple hunting search algorithm is presented. In Section 17.3 improvements are given. The Levy flight procedure is introduced. In Section 17.4 a numerical example, the welded beam design problem, is taken into consideration and application of the hunting search algorithm is demonstrated.

17.2 Original HuS algorithm in brief

The original hunting search algorithm can be presented using pseudo-code of the algorithm. Steps of the method are also listed in the following.

17.2.1 Description of the original hunting search algorithm

In Algorithm 19 the pseudo-code of the original hunting search algorithm is given, which summarizes the routine.

Algorithm 19 Pseudo code of original HuS.

determine the D-th dimensional objective function **F**
determine the range of variability for each j-th dimension $[X_{i,j}^{min}, X_{i,j}^{max}]$
determine the HuS algorithm parameters such as hunting group size, maximum movement toward the leader, hunting group consideration rate, maximum and minimum values of arbitrary distance radius, convergence rate parameters and number of iterations per epoch.
randomly create hunting group which consists of N hunters.
for each i-th hunter X_i from group **do**
    Evaluate objective function
    Determine the leader hunter X_i which has the minimum objective function
end for
while termination condition not met **do**
    for each i hunter in the group **do**

for each dimension **do**
update the position $\mathbf{X_{i,j}}$ using formula
$X_{i,j} = X_{i,j} + r \times (MML)(X^L_{i,j} - X_{i,j})$
check if the newly created value $X_{i,j}$ is within the range $[X^{min}_{i,j}, X^{max}_{i,j}]$ if not correct it
end for
evaluate the hunter X_i with objective function
if objective function of X_i better than leader **then**
assign the hunter X_i to the leader X^L_i
end if
end for
for each i hunter in the group **do**
for each dimension **do**
apply position correction-cooperation between members with $\mathbf{X_{i,j}}$ using formula

$$x_i^{j'} \leftarrow \begin{cases} x_i^{j'} \in \{x_i^1, x_i^2, i = 1, \ldots, x_i^{HGS}\} & i = 1, \ldots, N \\ x_i^{j'} = x_i^j \pm Ra \; with \; probability \; (1 - HGCR) & j = 1, \ldots, HGS \end{cases}$$

check if the newly created value $X_{i,j}$ is within the range $[X^{min}_{i,j}, X^{max}_{i,j}]$ if not correct it
end for
evaluate the hunter X_i with objective function
if objective function of X_i better than leader **then**
assign the hunter X_i to the leader X^L_i
end if
end for
for each i hunter in the group **do**
for each dimension **do**
apply reorganizing the group with $\mathbf{X_{i,j}}$ using formula
$x'_i = x^L_i \pm r \times (x^{max}_i - x^{min}_i) \times \alpha(-\beta \times EN)$
check if the newly created value $X_{i,j}$ is within the range $[X^{min}_{i,j}, X^{max}_{i,j}]$ if not correct it
end for
evaluate the hunter X_i with objective function
if objective function of X_i better than leader **then**
assign the hunter X_i to the leader X^L_i
end if
end for
end while

In step 1, parameters of the algorithm are initialized. The algorithm has eight parameters that require initial values to be assigned. These are hunting group size (number of solution vectors in the hunting group, HGS), maximum movement toward the leader (MML), hunting group consideration

rate (HGCR) which varies between 0 and 1, maximum and minimum values of arbitrary distance radius (Ra^{max} *and* Ra^{min}), convergence rate parameters (α *and* β) and number of iterations per epoch (IE). Then, in step 2, hunting group is initialized. Based on the number of hunters (HGS), the hunting group matrix is filled with feasible randomly generated solution vectors. The values of objective function are computed for each solution vector and the leader is defined depending on these values. Step 3 is devoted to the generation of the new hunters' positions. New solution vectors $x' = \{x'_1, x'_2, \ldots, x'_n\}$ are generated by moving toward the leader (the hunter that has the best position in the group) as follows.

$$x'_i = x_i + r(MML)(x_i^L - x_i) \qquad i = 1, \ldots, n \tag{17.1}$$

where MML is the maximum movement toward the leader, r is a uniform random number [0,1] and x'_i is the position value of the leader for the *ith* variable. If the movement of a hunter toward the leader is successful, it stays in its new position. However, if the movement is not successful, i.e., its previous position is better than its new position, it comes back to the previous position. This results in two advantages. First, the hunter is not compared with the worst hunter in the group to allow the weak members to search for other solutions. They may find better solutions. The second advantage is that, for prevention of rapid convergence of the group the hunter compares its current position with its previous position; therefore, good positions will not be eliminated. In step 4, position correction-cooperation between members is performed. In order to conduct the hunt more efficiently, the cooperation among hunters should be modeled. After moving toward the leader, hunters tend to choose another position in order to conduct the 'hunt' more efficiently, i.e. better solutions. Positions of the hunters can be corrected in two ways; real value correction and digital value correction. In real value correction which is considered in the present study, the new hunter's position $x' = \{x'_1, x'_2, \ldots, x'_n\}$ is generated from HG, on the basis of hunting group considerations or position corrections, which is expressed as;

$$x_i^{j'} \leftarrow \begin{cases} x_i^{j'} \in \{x_i^1, x_i^2, i = 1, \ldots, x_i^{HGS}\} & i = 1, \ldots, N \\ x_i^{j'} = x_i^j \pm Ra \text{ with probability } (1 - HGCR) & j = 1, \ldots, HGS \end{cases} \tag{17.2}$$

For instance, the value of the first design variable for the *jth* hunter $x_1^{j'}$ for the new vector can be selected as a real number from the specified $HG(x_i^1, x_i^2, \ldots, x_i^{HGS})$ or corrected using the HGCR parameter (chosen between 0 and 1).

In the above equation, HGCR is the probability of choosing one value from the hunting group stored in the HG. It is reported that values of this parameter

between 0.1 and 0.4 produce better results. Ra is referred to as an arbitrary distance radius for the continuous design variable, which can be reduced or fixed during the optimization process. Through the former assumption, Ra can be reduced by use of the following exponential function.

$$Ra(it) = Ra_{min}(x_i^{max} - x_i^{min}) \exp\left(\frac{\ln\left(\frac{Ra_{max}}{Ra_{min}}\right) \times it}{itm}\right) \tag{17.3}$$

where it represents the iteration number, x_i^{min} and x_i^{max} are the maximum and minimum possible values for x_i. Ra^{max} and Ra^{min} denote the maximum and minimum of relative search radius of the hunter, respectively and itm is the maximum number of iterations in the optimization process.

In digital value correction, instead of using real values of each variable, the hunters communicate with each other by the digits of each solution variable. For example, the solution variable with the value of 23.4356 has six meaningful digits. For this solution variable, the hunter chooses a value for the first digit (i.e. 2) based on hunting group considerations or position correction. After the quality of the new hunter position is determined by evaluating the objective function, the hunter moves to this new position; otherwise it keeps its previous position. Next in step 5 the hunting group is reorganized. In order to prevent being trapped in a local optimum they must reorganize themselves to get another opportunity to find the optimum point. The algorithm does this in two independent conditions. If the difference between the values of the objective function for the leader and the worst hunter in the group becomes smaller than a preset constant $\varepsilon 1$ and the termination criterion is not satisfied, then the algorithm reorganizes the hunting group for each hunter. Alternatively, after a certain number of searches the hunters reorganize themselves. The reorganization is carried out as follows: the leader keeps its position and the other hunters randomly choose their position in the design space.

$$x_i' = x_i^L \pm rand \times (max(x_i) - min(x_i)) \times \alpha \exp(-\beta \times EN) \tag{17.4}$$

where, x_i^L is the position value of the leader for the i^th variable, r represents the random number between 0 and 1, x_i^{min} and x_i^{max} are the maximum and minimum possible values for x_i, respectively. EN counts the number of times that the hunting group has trapped until this step. As the algorithm goes on, the solution gradually converges to the optimum point. Parameters α and β are positive real values which determine the convergence rate of the algorithm. Finally, the process is terminated if the predetermined value of iterations is reached.

17.3 Improvements in the hunting search algorithm

In order to increase the chance of obtaining the best solution, the Levy flight procedure [2] is adopted to the simple hunting search algorithm. How we adopt this to the algorithm is that, this step is added to the iteration process as a new search for the better position of the hunters. A Levy flight is a random walk in which the steps are defined in terms of the step-lengths which have a certain probability distribution, with the directions of the steps being isotropic and random. Hence Levy flights necessitate selection of a random direction and generation of steps under the chosen Levy distribution. The Mantegna algorithm [2] is one of the fast and accurate algorithms which generate a stochastic variable whose probability density is close to Levy stable distribution characterized by arbitrary chosen control parameter α $(0.3 \leq \alpha \leq 1.99)$. Using the Mantegna algorithm, the step size λ is calculated as;

$$\lambda = \frac{x}{|y|^{1/\alpha}} \tag{17.5}$$

where x and y are two normal stochastic variables with standard deviation σ_x and σ_y which are given as;

$$\sigma_x(\alpha) = \left[\frac{\Gamma(1+\alpha)sin(\pi\alpha/2)}{\Gamma((1+\alpha)/2)\alpha 2^{(\alpha-1)/2}}\right]^{1/\alpha} and\ \sigma_y(\alpha) = 1\ for\ \alpha = 1.5 \tag{17.6}$$

in which the capital Greek letter Γ represents the gamma function that is the extension of the factorial function with its argument shifted down by 1 to real and complex numbers. That is if k is a positive integer. $\Gamma(k) = (k-1)!$ Once the step size λ is determined according to random walk with Levy flights, the hunters' new positions are generated as follows;

$$x_i' = x_i \pm \beta\lambda r(x_i - x_i^L) \tag{17.7}$$

where, $\beta > 1$ is the step size which is selected according to the design problem under consideration, r is random number from standard normal distribution.

17.4 Applications of the algorithm to the welded beam design problem

17.4.1 Problem description

A carbon steel rectangular cantilever beam design problem [3] is taken as an optimization problem. The geometric view and the dimensions of the beam

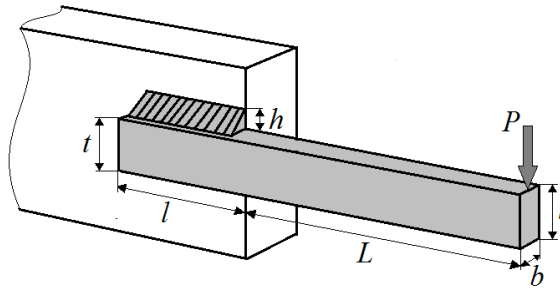

FIGURE 17.1
Welded beam.

are illustrated in Figure 17.1. The beam is designed to carry a certain P load acting at the free tip with minimum overall cost of fabrication. Geometric properties of the beam can be listed as the following: h the thickness of the weld, l is the length of the welded joint, t and b are the width and the thickness of the beam, respectively.

17.4.2 How can the hunting search algorithm be used for this problem?

We want to create an optimization problem to obtain the best dimensions for the welded beam. To do this, the objective function of the problem should be constructed first. This can be performed by use of the weight function of the beam. Then this weight function will be the objective function to be optimized or minimized. This function is given in the following.

$$f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \tag{17.8}$$

Once we have constructed this, now we can proceed with the design variables which are to be changed during the process. In fact, these are the dimensions of the beam listed above. We can determine the vector of variables as follows;

$x_1 : h$: the thickness of the weld
$x_2 : l$: the length of the welded joint
$x_3 : t$: the width of the beam
$x_4 : b$: the thickness of the beam

In the optimization applications, problems generally have constraints. Hence, an optimally designed welded beam should satisfy the below constraints for a correct design. These can be either side constraints or behavioral constraints. Below the first seven constraints are treated as behavioral constraints for this problem. And side constraints are given at the end.

$g_1(x) = \tau(x) - \tau_{max} \leqslant 0$: shear stress
$g_2(x) = \sigma(x) - \sigma_{max} \leqslant 0$: bending stress in the beam
$g_3(x) = x_1 - x_4 \leqslant 0$: side constraint
$g_4(x) = 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5 \leqslant 0$: side constraint
$g_5(x) = 0.125 - x_1 \leqslant 0$: side constraint
$g_6(x) = \delta(x) - \delta_{max} \leqslant 0$: end deflection of the beam
$g_7(x) = P - P_c(x) \leqslant 0$: buckling load on the bar

where

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2}\ x_1\ x_2}$$

$$\tau'' = \frac{MR}{J}$$

$$M = P\left(L + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_2}{2}\right)}$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$

$$\delta(x) = \frac{4PL^3}{Ex_3^3x_4}, \sigma(x) = \frac{6PL}{x_4x_3^2}$$

$$P_c(x) = \frac{4.013\ E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_2}{2L}\sqrt{\frac{E}{4G}}\right)$$

$P = 600\ lb, L = 14\ in,$
$E = 30 \times 10^6\ psi, G = 12 \times 10^6\ psi$
$\tau_{max} = 13.600\ psi, \sigma_{max} = 30.000\ psi,$
$\delta_{max} = 0.25\ in.$

The side constraints for the design variables are given as follows:

$0.1 \leqslant x_1 \leqslant 2.0, \qquad 0.1 \leqslant x_2 \leqslant 10$
$0.1 \leqslant x_3 \leqslant 10, \qquad 0.1 \leqslant x_4 \leqslant 2.0$

The iteration process is initialized by selecting the parameters. Then hunters are generated randomly. This means coordinates of the positions of each hunter are assigned. After that, it is observed whether the designs satisfy the constraints or not. If yes, then this design can be treated as a candidate. Then the objective function of each design is determined. Afterwards, the positions, namely the dimensions of the welded beam, are updated through the leader. That means new designs are related with the previous one of each hunter. Once again, it is observed whether the new designs are feasible or not. If yes, and weights are less than the previous one, then positions are updated and a new leader is determined. In order to obtain better solutions, positions of each hunter are corrected after moving towards the leader. While doing these changes, the algorithm generates random numbers. As conducted in the previous step, the leader is determined, which is also treated as the current best. In

the same manner, the update of the objective function for each hunter will be performed in the next two steps as well. Finally the design with the minimum objective function will be treated as the optimum solution of the problem.

17.4.3 Description of experiments

The method presented above was tested with the well-known benchmark 'welded beam design problem', the description of which is given in the previous part. Design pool was constructed by taking into account the interval of each variable. At the beginning of the solution process, we did not consider the value of the objective function of the problem. What is important at this stage is the generation of a predetermined number of feasible solutions. A flyback mechanism [4] was utilized in handling the constraints. While performing this, error is assumed to be 0. However, when the algorithm was in the iteration stage, an adaptive error strategy was applied. This strategy works in this way: If some hunters are slightly infeasible, then such hunters are kept in the solution. These hunters having one or more slightly infeasible constraints are utilized in the design process that might provide a new hunter that may be feasible. This is achieved by using larger error values initially for the acceptability of the new design vectors and then reducing this value gradually during the design cycles using finally an error value of 0.001 or whatever necessary value is required to be selected for the permissible error term towards the end of iterations. As expected, the algorithm produced a number of slightly feasible solutions and these solutions were kept in the memory as current best for the corresponding cycle. Afterwards, while iterations were proceeding, error was gradually reduced and finally feasible solutions were obtained in this neighborhood. The value of algorithm parameters are as follows: Number of hunters = 40, MML = 0.005, HGCR = 0.3, Ramax= 0.01, Ramin = 0.0000001, α = 1.2, β = 0.02, maximum number of iterations in one epoch = 25.

17.4.4 Result obtained

Optimum welded beam design was achieved after 119000 iterations with the objective function value of 1.724941. Design variables, e.g., the dimensions of optimum beam are 0.205731, 3.47112, 9.036624 and 0.205730. (Table 17.1)

The design history curve for these solutions is also plotted in Fig. 17.2, which displays the variation of the feasible best design obtained so far during the search versus the number of designs sampled. It is clear from this figure that the hunting search with levy flights algorithm performs the best convergence rate toward the optimum solution.

This solution was determined by the improved hunting search algorithm. On the other hand, the one attained by the simple hunting search method is

TABLE 17.1
Optimum designs obtained by simple and improved versions of hunting search algorithm.

Design Variable	**Method**	
	Improved HuS	**Simple HuS**
$\mathbf{x}_1$	0.205731	0.20573
$\mathbf{x}_2$	3.47112	3.47112
$\mathbf{x}_3$	9.036624	9.03663
$\mathbf{x}_4$	0.205730	0.20573
f	**1.724941**	**1.724944**
Number of iterations	**119000**	**119500**

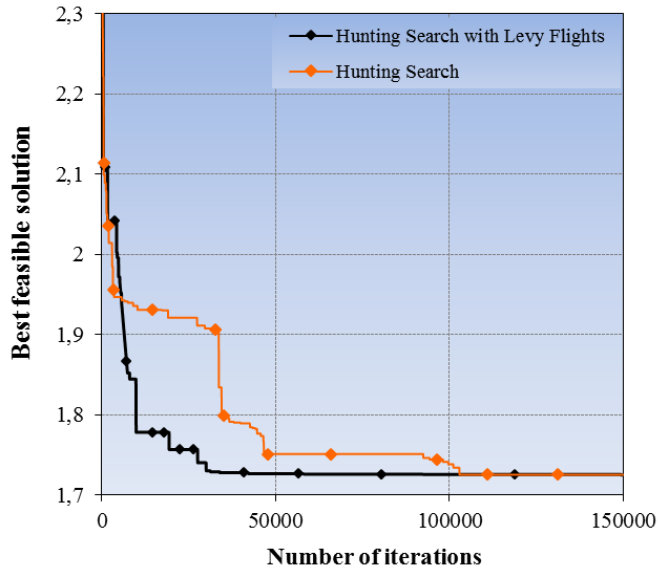

FIGURE 17.2
Design history graph for welded beam design problem.

stored as 1.724944. Although the difference between these two results seems to be negligibly small, this means a lot for the comparison. Therefore, in view of the results one can conclude that the Levy flight extension improved the performance of the algorithm.

17.5 Conclusions

In this subsection, improved and original version of hunting search optimum design algorithm presented in the previous sections is used to design well-known welded beam benchmark problem. Problem descriptions and the steps of the working process of the algorithm were given in detail. Dimensions of welded beam are treated as design variables of the optimum design problem and weight of the beam is assumed to be the objective function to be minimized. This problem was solved with both algorithms a number of times and results were compared. Accordingly, as compared to the solution of the standard hunting search algorithm, which is 1.724944, a much better final design value of 1.724941 is located by the hunting search with levy flights (Table 17.1). It is observed that Levy flights increases the performance of the algorithm. One common drawback of metaheuristic search methods is that the iteration process generally gets stuck in the local optima. In this study, it is observed that solutions with a constant error strategy experience this fact. However, it is observed in this study that an adaptive error strategy for constraint handling prevents this problem and provides an efficient search for each algorithm.

References

1. R. Oftadeh, M.J. Mahjoob, M. Shariatpanahi. "A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search". *Computers and Mathematics with Applications*, vol. 60(7), pp. 2087-2098, 2010.
2. R.N. Mantegna. "Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes". *Physical Review E.*, vol. 49(5), pp. 4677-4683, 1994.
3. K.M. Ragsdell, D.T. Phillips. "Optimal design of a class of welded structures using geometric programming". *Journal of Engineering for Industry*, vol. 98(3), pp. 1021-1025, 1976.
4. K. Deb. "Optimal design of a welded beam via genetic algorithms". *AIAA journal*, vol. 29(11), pp. 2013-2015, 1991.