TẠP CHÍ KHOA HỌC & CÔNG NGHỆ ĐẠI HỌC DUY TÂN
DTU Journal of Science and Technology 3(52) (2022) 34-40
TRƯỜNG ĐẠI HỌC DUY TÂN
DUYTAN UNIVERSITY

Particle Swarm Optimization using ε constraint-handling method developed in Python

Thuật toán tối ưu hóa bầy đàn sử dụng phương pháp xử lý ràng buộc ε được phát triển với Python

Hoang Nhat Duc[a,b*], Nguyen Quoc Lam[a,b]
Hoàng Nhật Đức[a,b*], Nguyễn Quốc Lâm[a,b]

[a]*Institute of Research and Development, Duy Tan University, Da Nang, 550000, Vietnam*
[a]*Viện Nghiên cứu và Phát triển Công nghệ Cao, Đại học Duy Tân, Đà Nẵng, Việt Nam*
[b]*Faculty of Civil Engineering, Duy Tan University, Da Nang, 550000, Vietnam*
[b]*Khoa Xây dựng, Trường Đại học Duy Tân, Đà Nẵng, Việt Nam*

(Ngày nhận bài: 25/10/2021, ngày phản biện xong:16/5/2022, ngày chấp nhận đăng:22/5/2022)

Abstract

This research work aims at implementing a swarm intelligence based approach for solving complex constrained optimization tasks. The ε Particle Swarm Optimization (ε PSO) is selected as the employed global optimizer. This optimization method is developed in Python to facilitate its implementations. The newly developed program has been tested with two basic design problems in civil engineering.

Keywords: Particle Swarm Optimization; Design optimization; Swarm intelligence; Metaheuristic; Civil engineering.

Tóm tắt

Nghiên cứu của chúng tôi xây dựng một công cụ tối ưu hóa dựa trên trí tuệ bầy đàn. Phương pháp ε PSO được lựa chọn để giải các bài toán tối ưu hóa toàn cục. ε PSO đã được chúng tôi phát triển với ngôn ngữ lập trình Python để đẩy mạnh tính ứng dụng của công cụ này trong thực tiễn. Chương trình mới đã được thử nghiệm với 2 bài toán tối ưu hóa cơ bản trong ngành xây dựng.

Từ khóa: Thuật toán tối ưu hóa bầy đàn; Tối ưu hóa thiết kế, Trí tuệ bầy đàn; Thuật toán tìm kiếm, Xây dựng dân dụng.

1. Introduction

Constrained optimization is an important research area in various engineering fields [1-5]. Civil engineers are required to solve design problems in which an objective function is either minimized or maximized and a set of constraints must be satisfied [6, 7]. Such design problems can be very challenging because they often involve a large number of design variables and constraints. Researchers and practitioners have increasingly relied on metaheuristic to deal with constrained optimization problems [8-15].

Notably, Takahama, Sakai and Iwane [16] has proposed the ε-method used with metaheuristics for dealing with constrained optimization problems. Using the ε method, the selection operation of metaheuristics is

*Corresponding Author: *Hoàng Nhật Đức*, Institute of Research and Development, Duy Tan University, Da Nang, 550000, Vietnam; Faculty of Civil Engineering, Duy Tan University, Da Nang, 550000, Vietnam.
Email: hoangnhatduc@duytan.edu.vn

modified by taking into account the constraint violation degree of each individual. Hence, this approach is capable of handling a large number of constraints. In this research, an optimization model based on the ε method and the Particle Swarm Intelligence metaheuristic is developed in Python. The newly developed tool is tested with two basic design tasks in civil engineering. We select Python in this work because it is an interpreted high-level general-purpose programming language that has the advantages of simplicity and code readability.

2. Methodology

Particle Swarm Optimization (PSO) [17] is confirmed to be one of the most widely and successfully used metaheuristic. PSO is robust and can easily be implemented on a wide range of optimization tasks [18-24]. This metaheuristic mimics the behavior of flocks of birds or pools of fish. The movement of a swarm is directed towards the optimization of food search. The movement of these animals is optimized based on their coordinated movements [25, 26].

The PSO algorithm first generates a swarm of S individual within the boundary of the searched space. Given an objective function and a set of constraints, the algorithm computes the objective function and constraint functions' values. Similar to the standard PSO, the εPSO also relies on the concept of local and global best. The local best of a particle is associated with a position (X_{LB}), an objective function value (F_{LB}), and a constraint satisfaction index (ϕ_{LB}). The global best is also characterized by these three records (X_{GB}, F_{GB}, and ϕ_{GB}). The constraint violation degree $\phi(x)$ can be coded in Python (see **Fig. 2.1a**) and is defined as follows [16, 27, 28]:

$$\phi(x) = \sum_j |\min_j(0, g_j(x))| + \sum_j \max_j |h_j(x)| \quad (1)$$

```
for i in range(PS):
    Pop_i = Pop[i, :]
    Fval[i], G_i = ObjFun(Pop_i)
    Nc = G_i.shape[0] # number of constraints
    Sum_Constr_Vio_i = 0
    for k in range(Nc):
        if G_i[k] < 0:
            Sum_Constr_Vio_i = Sum_Constr_Vio_i + abs(G_i[k])
    Phi[i] = Sum_Constr_Vio_i
```

(a)

```
def EpsCompare(F0 = 100, Phi0 = 1.4, F1 = 1, Phi1 = 2, esp = 1):
    Winner = 1
    if Phi0 <= esp and Phi1 <= esp and F0 < F1:
        Winner =  0
    if Phi0 == Phi1 and F0 < F1:
        Winner =  0
    if Phi0 > esp or Phi1 > esp:
        if Phi0 < Phi1:
            Winner =  0
    return Winner
```

(b)

Fig. 2.1 The quantification of the constraint violation and ε selection operation coded in Python

Using the computed values of $\phi(x)$, the ε selection operation can be coded in Python (refer to **Fig. 2.1b**) and stated as follows:

$$(f_1,\phi_1) <_{\varepsilon} (f_2,\phi_2) = \begin{cases} f_1 < f_2 \text{ if } \phi_1,\phi_2 \le \varepsilon \\ f_1 < f_2 \text{ if } \phi_1 = \phi_2 \\ \phi_1 < \phi_2, \text{otherwise} \end{cases} \quad (2)$$

where ε is initially set to be $\phi(x_N)$ and N = 0.2S. Moreover, ε is gradually reduced and subsequently set to be 0 if the iteration counter g > 0.7G_{Max} (G_{Max} = the maximum number of searching iterations).

```python
# Update velocity
for d in range(D):
    r1 = rn.random()
    r2 = rn.random()
    V[i, d] = K*(V[i, d] +
                 c1*r1*(Local_Best_Pop[i,d]-Pop[i,d]) +
                 c2*r2*(Global_Best_Sol[d]-Pop[i,d]))
# update position and evaluate
for d in range(D):
    Pop[i,d] = Pop[i,d] + V[i,d]

    if Pop[i,d] > UB[d]:
        Pop[i,d] = UB[d] -  (UB[d]-LB[d])*rn.random()/10
    if Pop[i,d] < LB[d]:
        Pop[i,d] = LB[d] +  (UB[d]-LB[d])*rn.random()/10
```

Fig. 2.2 The velocity computation and position update coded in Python

In a searching iteration, the velocity of a particle can be coded in Python (refer to **Fig. 2.2**) and stated as as follows:

$$v_{i,d} == K[v_{i,d} + c_1 \times r_1() \times (x_{LB,d} - x_{id}) + c_2 \times r_2() \times (x_{GB,d} - x_{id})] \quad (3)$$

where $K = \dfrac{2}{|2 - U - \sqrt{U^2 - 4U}|}$ is the constriction factor and U = c_1 + c_2. $v_{i,d}$ is the velocity of the ith particle in dth dimension. $x_{LB,d}$ and $x_{GB,d}$ is the local best and global best of x. r_1 and r_2 are two uniform random number within [0,1].

Based on the computed velocity, the new position of the particle is computed as follows:

$$x_{i,d} = x_{i,d} + v_{i,d} \quad (4)$$

3. Experimental result

In this section, the εPSO, which is coded in Python, is employed to solve two basic constrained optimization problems in construction engineering. The first problem involves designing a bar of different cross-sections is subjected to a tensile force F = 50kN (see **Fig. 3.1**). The design variables include lengths (L_1, L_2, and L_3) and diameters (D_1, D_2, and D_3) of three sections of the bar. The objective function is the total volume of the bar. This problem has three constraints involving the stress in each section and the total elongation of the whole system. The problem is coded in Python (refer to **Fig. 3.2**) and mathematically stated as follows:

$$\text{Min} \quad f = \sum_{i=1}^{3} L_i \times D_i \quad (5)$$

s.t. $G_1(x) = 35 - F/A_1 \ge 0$

$G_2(x) = 150 - F/A_2 \ge 0$

$G_3(x) = 635 - F/A_3 \ge 0$

$G_4(x) = 0.15 - \left(\dfrac{FL_1}{A_1E} + \dfrac{FL_2}{A_2E} + \dfrac{FL_3}{A_3E}\right) \ge 0$

where $G_1(x)$, $G_2(x)$, $G_3(x)$, $G_4(x)$, and $G_5(x)$ are the problem's constraints.

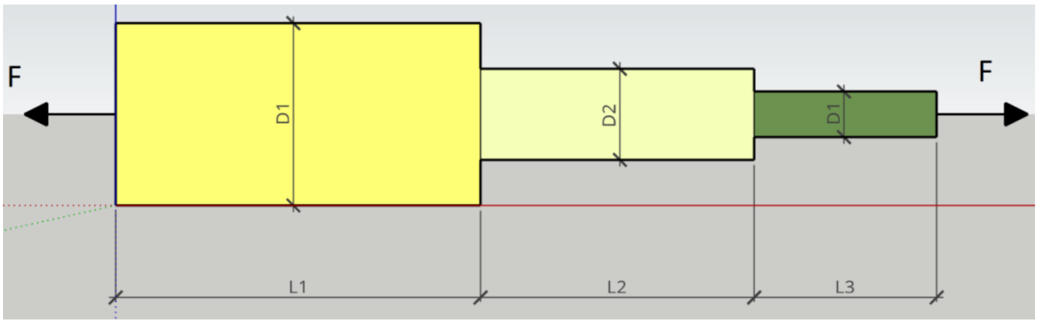


Fig. 3.1 Illustration of optimization problem 1

```python
def ConObjFun_Problem1(X = np.array([1, 6])):
    L = np.array([(X[0]), X[1], X[2]]) # mm
    D = np.array([X[3], X[4], X[5]]) # mm
    E = 200*(10**3) # N/mm2
    F = 50 *(10**3) # N
    Ne = 3 # number of elements
    Area = np.zeros(Ne)
    for i in range(Ne):
        Area[i] = np.pi * (D[i]**2)/4
    Stress = np.zeros(Ne)
    for i in range(Ne):
        Stress[i] = F/Area[i]
    # print("Stress:", Stress)
    # dL = PL/(AE)
    dL = np.zeros(Ne)
    for i in range(Ne):
        dL[i] = F*L[i]/(Area[i]*E)
    Total_L = np.sum(dL)

    f = X[0]*X[3] + X[1]*X[4] + X[2]*X[5]
    g = np.zeros(4)
    g[0] = 35 - Stress[0]
    g[1] = 150 - Stress[1]
    g[2] = 635 - Stress[2]
    g[3] = 0.15 - Total_L
    return f, g
```

Fig. 3.2 Optimization problem 1 coded in Python

Fig. 3.3 Illustration of optimization problem 1

```python
def ConObjFun_Problem2(X = np.array([1, 3])):
    n = round(X[0], 0) # 2 - 10
    b = X[1] # 0.1 - 0.5
    d = X[2] # 0.1 - 0.5
    L0 = 4.8 # m
    L1 = 3.0 # m
    s = L0/(n-1)
    w = s*6390
    Mmax = w*(L1**2)/8
    Smax = w*L1/2
    Tmax = 3*Smax/(2*b*d)
    f = d*b*L1*400
    sig_allow = 10* 1000000
    T_allow = 0.448 *  1000000
    E = 1600000 *  0.00689476 *  1000000
    I = b*(d**3)/12
    delta_allow = L1/360
    g = np.zeros(4)
    A = b*d*d/6
    g[0] = sig_allow - Mmax/A
    g[1] = T_allow - Tmax
    g[2] = delta_allow - 5*w*(L1**4)/(384*E*I)
    g[3] = s - 1.5
    return f, g
```

Fig. 3.4 Optimization problem 2 coded in Python

The second problem involves designing a system of wood beam supporting concrete slab formwork (demonstrated in **Fig. 3.3** and coded in Python as shown in **Fig. 3.4**). The decision variables are the cross-sectional parameters (the depth d and the width b) and the number of required beams n. Thus, the beams are required to support the operation of constructing a reinforced concrete slab structure. The objective herein is to find a set of d, b, and n which minimizes the material cost of the beams.

The centre-to-centre spacing of the beams s is as follows:

$$s = L_0 / (n-1) \tag{6}$$

where L_0 = 4.8m.

The load per unit length acting on the slab formwork is given by:

$$w = s \times \lambda \tag{7}$$

where λ = 6390N/m^2 denotes the load caused by concrete weight that acts on the slab formwork.

The maximum bending moment in the beams caused by w is computed as follows:

$$M_{max} = wL1^2 / 8 \tag{8}$$

The maximum shear forced in the beams caused by w is computed as follows:

$$S_{max} = w \times L1 / 2 \tag{9}$$

The maximum shear stress in the beams caused by w is computed as follows:

$$T_{max} = 3S_{max\,2} / (2b \times d)$$

The constraints of this problem specify limitations on (i) bending stress, (ii) shear stress, (iii) deflection of the beams [29-31], and the requirement for s to ease on-site movement. Hence, this problem is mathematically formulated as follows:

$$Min. f = d \times b \times L1 \times \gamma_{Wood} \tag{10}$$

s.t.

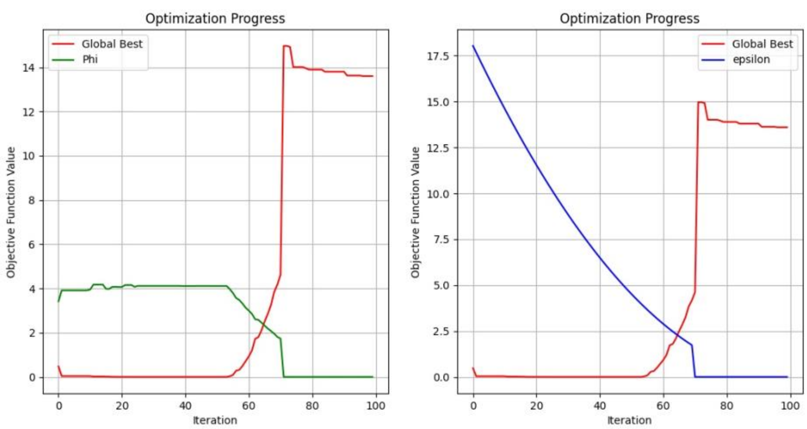
$$G_1(x) = \sigma_{Allow} - \frac{M_{max}}{(bd^2/6)} \geq 0$$

$$G_2(x) = \tau_{Allow} - T_{max} \geq 0$$

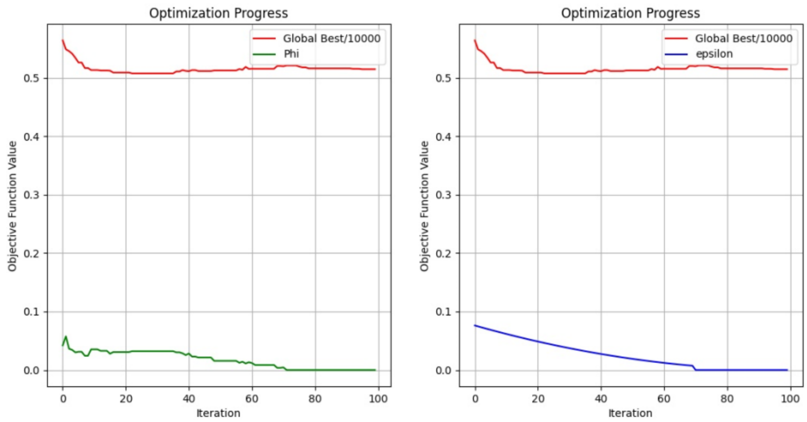
$$G_3(x) = \delta_{Allow} - 5wL1^4 / (384\,\text{EI}) \geq 0$$

$$G_4(x) = s - 1.5 \geq 0$$

where L_1 = 3 m is the length of a beam. Mass density of wood γ_{Wood} is 400 kg/m^3. σ_{Allow} = denotes the allowable bending stress = 10000000 N/m^2. τ_{Allow} =0.448 × 1000000 N/m^2. The modulus of elasticity of wood E = 1600000 × 0.00689476 × 1000000 N/m^2. The moment of inertia of the cross section about the centroidal axis I = bd^3/12. $\delta_{Allow} = L1/360$.



(a)



(b)

Fig. 3.5 Optimization process: (a) Problem 1 and (b) Problem 2

The optimization progress of the two problems solved by εPSO is demonstrated in **Fig. 3.5**. Herein, the objective function value, the constraint violation degree ϕ (Phi), and ε (epsilon) parameter are plotted. It is noted that when ε = 0, the allowance for constraint violation completely stops. For problem 1, the best found solution X = [80.08, 60.49, 40.37, 42.65, 20.62 12.01] with the objective function value = 5148.29. The constraint vector of this problem with the found solution is G = [3.46e-03, 3.65e-01, 1.94e+02, 1.70e-03]. All elements of G are greater than 0 and this indicates that all of the constraints are satisfied. For the problem 2, the best found solution is [4, 0.20, 0.26] with the objective function = 61.63. The constraint vector of the 2nd problem with the found solution is G = [4.79e+06 6.86e+01 4.90e-03 1.00e-01].

4. Conclusion

This research develops a metaheuristic based approach based on the PSO metaheuristic and the ε method for constraint handling. This hybrid approach has been constructed in Python to ease its implementation and development of optimization models. The program, named as εPSO, has been tested with two basic constrained optimization tasks in which a bar with different cross-section and a system of wood beam supporting concrete slab formwork are designed. Experimental result shows that εPSO is a capable method to assist civil engineers in the tasks of design optimization.

References

[1] J.S. Arora (2016), Introduction to Optimum Design, Fourth Edition, Academic Press.

[2] D. Goldberg (1989), Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Professional, ISBN 978-0201157673.

[3] M.-Y. Cheng, D.-H. Tran, N.-D. Hoang (2017) Fuzzy clustering chaotic-based differential evolution for resource leveling in construction projects, Journal of Civil Engineering and Management, 23 113-124.

[4] N.D. Hoang (2020) Metaheuristic based resource leveling using CPM project scheduling software program developed in .NET framework, DTU Journal of Science and Technology, 02 22-27.

[5] H.-H. Tran, N.-D. Hoang (2014) A Novel Resource-Leveling Approach for Construction Project Based on Differential Evolution, Journal of Construction Engineering, 2014 7.

[6] S.M. Nigdeli, G. Bekdaş, X.-S. Yang (2018) Metaheuristic Optimization of Reinforced Concrete Footings, KSCE Journal of Civil Engineering, 22 4555-4563.

[7] N.D. Hoang (2019) FR-DE Excel Solver: Differential Evolution with Deb's feasibility rules for solving constrained optimization problems in civil engineering, DTU Journal of Science and Technology 04 (35).

[8] X.-S. Yang (2014), Nature-Inspired Optimization Algorithms, Elsevier.

[9] C.A.C. Coello (2018) Constraint-handling techniques used with evolutionary algorithms, Proceedings of the Genetic and Evolutionary Computation Conference Companion, ACM, Kyoto, Japan, pp. 773-799.

[10] C.A. Coello Coello (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, Computer Methods in Applied Mechanics and Engineering, 191 1245-1287.

[11] K. Deb (2000) An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering, 186 311-338.

[12] N.-D. Hoang (2014) NIDE: A Novel Improved Differential Evolution for Construction Project Crashing Optimization, Journal of Construction Engineering, 2014 7.

[13] N.Đ. Hoàng, Q.L. Nguyễn, Q.N. Phạm (2015) Tối ưu hóa tiến độ và chi phí cho dự án xây dựng sử dụng thuật toán tiến hóa vi phân, Tạp Chí Khoa Học và Công Nghệ, Đại Học Duy Tân, 1 135–141.

[14] N.Đ. Hoàng, D.T. Vũ (2015) Tối ưu hóa kết cấu có điều kiện ràng buộc sử dụng thuật toán bầy đom đóm và các hàm phạt, Tạp Chí Khoa Học và Công Nghệ, Đại Học Duy Tân, 2 75–84.

[15] H. Nhat-Duc, L. Cong-Hai (2019) Sử dụng thuật toán tiến hóa vi phân cho các bài toán tối ưu hóa kết cấu với công cụ DE-Excel solver, DTU Journal of Science and Technology, 03 97-102.

[16] T. Takahama, S. Sakai, N. Iwane (2006) Solving Nonlinear Constrained Optimization Problems by the ε Constrained Differential Evolution, 2006 IEEE International Conference on Systems, Man and Cybernetics, pp. 2322-2327.

[17] J. Kennedy, R. Eberhart (1995) Particle swarm optimization, Neural Networks, 1995. Proceedings., IEEE International Conference on, pp. 1942-1948 vol.1944.

[18] A. Banks, J. Vincent, C. Anyakoha (2007) A review of particle swarm optimization. Part I: background and development, Natural Computing, 6 467-484.

[19] I.-T. Yang (2007) Using Elitist Particle Swarm Optimization to Facilitate Bicriterion Time-Cost Trade-Off Analysis, Journal of Construction Engineering and Management, 133 498-505.

[20] M.R. Bonyadi, Z. Michalewicz (2017) Particle Swarm Optimization for Single Objective Continuous Space Problems: A Review, Evolutionary Computation, 25 1-54.

[21] P.-T. Ngo, N.-D. Hoang, B. Pradhan, Q. Nguyen, X. Tran, Q. Nguyen, V. Nguyen, P. Samui, D. Tien Bui (2018) A Novel Hybrid Swarm Optimized Multilayer Neural Network for Spatial Prediction of Flash Floods in Tropical Areas Using Sentinel-1 SAR Imagery and Geospatial Data, Sensors, 18 3704.

[22] D. Tien Bui, N.-D. Hoang, V.-H. Nhu (2018) A swarm intelligence-based machine learning approach for predicting soil shear strength for road construction: a case study at Trung Luong National Expressway Project (Vietnam), Engineering with Computers.

[23] D. Tien Bui, V.-H. Nhu, N.-D. Hoang (2018) Prediction of soil compression coefficient for urban housing project using novel integration machine learning approach of swarm intelligence and Multi-layer Perceptron Neural Network, Advanced Engineering Informatics, 38 593-604.

[24] H. Nguyen, N.-M. Nguyen, M.-T. Cao, N.-D. Hoang, X.-L. Tran (2021) Prediction of long-term deflections of reinforced-concrete members using a novel swarm optimized extreme gradient boosting machine, Engineering with Computers.

[25] A.Q.H. Badar (2021) Evolutionary Optimization Algorithms, CRC Press.

[26] N.-D. Hoang, D.-T. Nguyen, X.-L. Tran, T.-M.-T. Nguyen (2018) Ứng dụng thuật toán bầy đàn cho các bài toán tối ưu hóa trong quản lý xây dựng với công cụ PSO-Excel solver, Tạp Chí Khoa Học Công Nghệ, Đại Học Duy Tân, 5.

[27] T. Takahama, S. Sakai, Solving Difficult Constrained Optimization Problems by the ε Constrained Differential Evolution with Gradient-Based Mutation, in: E. Mezura-Montes (Ed.) Constraint-Handling in Evolutionary Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 51-72.

[28] T. Takahama, S. Sakai, N. Iwane (2005) Constrained Optimization by the ε Constrained Hybrid Algorithm of Particle Swarm Optimization and Genetic Algorithm, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 389-400.

[29] J.M. Gere, B.J. Goodno (2013), Mechanics of Materials, SI Edition, Cengage Learning.

[30] J.L. Meriam , L.G. Kraige , J.N. Bolton (2016), Engineering Mechanics: Statics, John Wiley & Sons.

[31] T.A. Philpot, J.S. Thomas (2018), Mechanics of materials an integrated learning system, Wiley.