International Journal of Applied Science and Engineering

# Solving unbounded knapsack problem using evolutionary algorithms with bound constrained strategy

**Vani Suthamathi Saravanarajan[1], Rung-Ching Chen[1*], Christine Dewi[1, 2], Long-Sheng Chen[1]**

[1] *Department of Information Management, Chaoyang University of Technology, Taichung, Taiwan, R.O.C.*
[2] *Faculty of Information Technology, Satya Wacana Christian University, Central Java, Indonesia*

## ABSTRACT

Unbound Knapsack Problems (UKP) are important research topics in many fields like portfolio and asset selection, selection of minimum raw materials to reduce the waste, and generating keys for cryptosystems. Given the uncertainty in data, capacity, and time constraints, users have to look at the possible combination of data to get maximum benefit. This paper uses UKP as a numerical model to represent different industrial combination problems. It applies Evolutionary Algorithms (EA) with Bound Constrained Strategy (BCS) to construct a search space and algorithm parameters for finding the optimal solution. Evolutionary Algorithms (EA) like Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) are designed based on reusable components for the algorithms to converge faster. Simulation for various objectives indicates that the GA and PSO can find the near-optimal solution in all cases. The execution time of GA and PSO for different goals and the variations in the algorithm parameters are measured. The measurement result shows the performance of GA and PSO is the same on an average for the differences in bounded constraints and parameter settings.

***Keywords:*** Unbound knapsack problem, Constrained optimization, Genetic algorithm, Particle swarm optimization, Evolutionary algorithms.

**OPEN ACCESS**

**Received:** August 29, 2020

**Accepted:** September 26, 2020

**Corresponding Author:**
Rung-Ching Chen
crching@cyut.edu.tw



**Publisher:**
Chaoyang University of Technology
**ISSN:** 1727-2394 (Print)
**ISSN:** 1727-7841 (Online)

## 1. INTRODUCTION

Evolutionary Algorithms (EA) are generally perceived as fruitful optimization techniques with regards to unconstrained streamlining. The use of EA to constrained optimization has lately picked up attention. Looking for an ideal solution in the search space with certain constraints on the optimization tasks led to the development of the constrained optimization techniques. In many optimization problems, constraints result from physical boundaries on the input data, time limit, problem-specific considerations, and limitations on the problem resources. This is especially obvious with regards to black-box and random input based optimization techniques. The conceptual perspective of EA is very less matured. Subsequently, the development of EA techniques on constrained optimization issues has become popular among researchers. Few applications of constrained optimization are finding the collision probability between robot and obstacle using a chance-constrained nonlinear model of predictive control problem (Zhu and Alonso-Mora, 2019), Delay-Optimal Joint Processing in Computation-Constrained Fog Radio Access Networks (Han et al., 2019), Sensor determination technique that limits the input sample size, subject to the requirements on the error probabilities and sensor uses (Li et al., 2019), Stochastic Spacecraft Trajectory Optimization with the Consideration of Chance Constraints (Chai et al., 2020) and so on.

Unbound Knapsack Problem (UKP) is a search constrained optimization problem emerging from many real-world problems like budget allocation, project management, and traffic scheduling, etc. There are restrictions to the physical bounds of input data, limited resources, time, and solution parameters. In this paper, UKP is defined as the problem of finding the set of items $I_s$ in the search space $S$ of N-dimensional real coordinate space $R^N$. The objective function $f(I_s)$or error function to find the feasible solution for the UKP is defined in Equations (1)-(3).

Minimize: $f(I_s)= C - \sum_{i=1}^{n} NC_i \times W_i$ (1)

Maximize : $\sum_{i=1}^{n} NC_i \times V_i$ (2)

$I_s \subseteq S \subseteq R^N$ (3)

where $C$ is the maximum capacity of Knapsack, $NC_i$ is the number of copies of item *i*, $W_i$ is the weight of item *i* and $V_i$ is the value of item *i* respectively and $1 \leq i \leq n$.

Recently, optimization techniques have incorporated evolutionary algorithms with adaptive strategies. Evolutionary algorithms with structure mutations are proposed in Yuan et al. (2015). Optimization is the process of obtaining the best outcome under a given set of circumstances. Many population-based search techniques have surfaced to become a mainstay of optimization. These techniques rely on search ideologies that work based on manipulating samples, which are representatives of the search sub-regions within the solution landscape (Chen et al., 2011).

Constrained optimization models are developed to find the best solution, either minimize or maximize the goal specified in the objective function and helping the decision-makers to take the best action in a reasonable time. Some of the challenges of dynamic constrained optimization are discussed in Nguyen and Yao (2012). The optimization technique is an important research topic when dealing with the uncertainty of the data. The design of an optimal controller to minimize the integral of squared error (ISE) of the closed-loop system for an interval plant via evolutionary approaches is proposed in Hsu and Yu (2004). The advent of modern computing technologies has enhanced the size and complexity of optimization problems that can be solved in a reasonable time.

Summarizing, the main contribution of this paper is fourfold: (1) To utilize the advantages of EA to find the best optimal solution in a reasonable time using bound constrain strategies. (2) To build a reusable EA procedure for solving two different types of UKP based real-world problem. (3) To present a system implementation, based on the proposed strategy. (4) Finally, to evaluate scalability and the computational cost of EAs with two optimization objectives, namely, coverage by minimum error and maximum profit.

## 2. RELATED WORKS

Advancement in Machine Learning in recent years has contributed to several methods for solving real-world problems. The quest for an optimal solution for the real-world problems is based on the choice of decision variables and restricted to unique constraints of the search space for the solution. UKP refers to many real-world problems like group seat reservation of knapsack problems (Deplano et al., 2019); many researchers and engineers work to find the best alternative. UKP can also be included in the Multiobjective Optimization Problem (MOP), with two or more objectives to solve. Many researchers in the past were involved in solving the MOP, but the metaheuristic based Evolutionary Algorithms have gained popularity in recent years. Despite the intense research activities, there are a few open challenges to the algorithm designs and the scalability of the objective functions, as discussed in Coello et al. (2019).

Different optimization algorithms are used in the current industry like Teaching Learning Based Evolutionary Algorithms (EA) for finding the optimal global solution (Satapathy et al., 2013). EA is an artificial intelligence technique for finding the optimal solution inspired by natural evolution. In this paper, GA and PSO are considered to solve the UKP problem. PSO is applied in Nonlinear Optimization Problems, Training Neural Networks, Heating System Planning, and Power Systems. Hellwig et al. (2019) reviewed EA in constrained benchmarking and shed light on solving problems in different domains. Genetic Algorithms are applied in different system modeling, where the experimental data is iterated multiple times to get the best optimal solutions like the one discussed by Marius et al. (2017).

Though all real-world problems look complicated, they are bound by certain patterns and constraints. In some evloutionary constrained optimization techniques, the values for bounds are fixed and predefined monotonically by a non-decreasing sequence of values in the search space. This approach was suitable for many problems; it worked well for simple problems with limited data sets but failed for the more difficult ones with large data sets. There was a need to develop an adaptive approach for setting the constrained parameter values dynamically by the EA itself. For example, initial population information can be computed adaptively, leading to different population values for various problems. T Efren Ju'arez-Castillo et al. (2017) discusses Bounded strategies or Constrained Optimization techniques in PSO. Jiang, H et al.(2019) used Least angle regression algorithm to reduce the high-dimensional data. excellent performance, especially for high-dimension data.

By surveying EAs applied to optimization problems, it is found that the existing techniques lack the reuse of knowledge acquired from one problem to another. Notably, the major drawback of existing search methods is the assumption of non-similarities between the current problem and those encountered in the past. In this paper, more focus is given for developing a common procedure, which is applied to different types of EAs. The reusability of the constrained computation narrowed the search space, hence, helping to reduce the computational cost and the convergence of data much faster. In the study of Chen et al. (2011), the assumption of zero usable information and the lack of knowledge transfers across the problems is discussed in detail.

## 3. METHODOLOGY

The methodology used to solve the UKP optimization problem is the modified EA with the bound-constrained strategy applied in search space, initial population, and evolutionary process. When the constructed system is constrained, the objective function's error stays very minimal, and the data converges to an optimized solution faster. The desired performance for the robust control systems is achieved using the constraint-following error is discussed in Sun et al. (2020). Based on the literature reviews, boundary constrained strategies are considered for optimization problems, which means the search space $S$ consists of n dimensional real-valued parameters $r$, where each parameter is bounded to an interval [LowerBound(LB), UpperBound(UB)] as referred in Nguyen and Yao (2012). The bound constraint coefficient for each parameter is $\beta_i$ and $\gamma_i$, defined in Equations (6) and (7) respectively.

$$S \subseteq r^n \quad (4)$$

$$r = \mathrm{C}/\max(W) \quad (5)$$

$$\gamma_i \in \{ 0, \max(V)/V_i \} \quad i = 1,\ldots\ldots,n \quad (6)$$

$$\beta_i \in \{ 0, \ \mathrm{C}/W_i \} \quad i = 1,\ldots\ldots,n \quad (7)$$

The objective function defined for UKP in Equations (1) and (2) are modified to Equations (8) and (9) as shown below:

$$\text{Minimize: } f(I_s) = \mathrm{C} - \textstyle\sum_{i=1}^{n} W_i \times \beta_i \quad (8)$$

$$\text{Maximize: } \textstyle\sum_{i=1}^{n} V_i \times \beta_i \quad (9)$$

The EA techniques based on bounded strategy are developed based on the optimization problem defined by the Equations (4)-(9). From the literature reviews, it is evident that different EAs like PSO and GA share many common points though they are based on two different evolutionary principles. Both algorithms start with the initial population, have fitness values to evaluate the population, update the population and search for the optimum solution with their own evolutionary techniques.

1. Creation of Search Space $S_n$ for n number of items is the union of bound constraint coefficients of the n items as shown in Equation (10).

$$S_n = \{ \beta_1 \cup \beta_2 \ \cup \ldots\ldots\ldots\ldots\ldots \cup \ \beta_n \} \quad (10)$$

2. Creation of Initial Population based on the Search Space and Equation (5) is given in Equation (11).

$$\text{Total Number of Individuals in Initial Population} = r \times \mathrm{n} \quad (11)$$

3. The initial population fitness is evaluated as a single function, i.e., the evaluation of the objective function and all related constraints, as shown in Chai et al. (2020).
4. The evolution process of the population is based on PSO and GA's evolution principles, but modifying certain steps based on the bound constraint coefficients $\beta_i$ and $\gamma_i$ .
5. The evolution process of the Dataset towards the optimal solution is continued until the termination criteria are met.

### 3.1 Genetic Algorithm

Genetic algorithms are used in many industrial processes to optimize the model parameters, as investigated by Marius et al. (2017). In this paper, the GA is modified based on the EA procedures developed with bound constraint coefficients. The creation of the Initial Population, the development of genetic mechanisms of selection, crossover, and mutations are driven by the bound constraint coefficients towards the feasible search space for finding the optimal solution. The flow diagram of the modified GA is shown in Fig. 1.

The creation of chromosomes and generation of the Initial Population involves identifying the gene, size of the population, and the length of the chromosomes using bound-constrained strategy. The chromosomes are made up of sequences of genes with the 0 and 1 binary combinations. The number of selection of an item is represented in the form of genes. For example, the number of selection of four items are bounded to the constraints 2, 0, 0 and 1 respectively, then the gene coding for the selection of the items are shown in Table 1 and the chromosome is represented as 10000001. The fitness of chromosomes are evaluated using the Equation (8) and Equation (9).

**Table 1.** Gene coding

| Item1 | | Item2 | | Item3 | | Item4 | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Finding an efficient crossover mechanism to recombine individuals into higher quality solutions plays an essential role in evolutionary computations, as discussed in Corus and Oliveto (2018). The Multi-point crossover method is used in this research. An N point crossover mates the selected parents based on the fitness values, and the length of the gene gives the benefit of N. Each bit in the chromosome is subject to mutation with a probability. Suppose a random float number is more significant than 0.5 flip 0 with one and vice versa. The maximum number of generations is considered as the stopping criteria.The creation of chromosomes with bound constrained strategy are shown in algorithm 1.

### 3.2 Particle Swarm Optimization

Particle Swarm Optimization (PSO), as discussed in Chai et al. (2020), is an evolutionary computation technique based on the initial population called swarm with the n possible solution candidates as particles. In PSO, a swarm of n particles (individuals) is guided towards the best optimal solution, either by communicating their best-known position or the entire swarm best position with one another using search directions. The objective functions guide the most important position of the particles. The bound

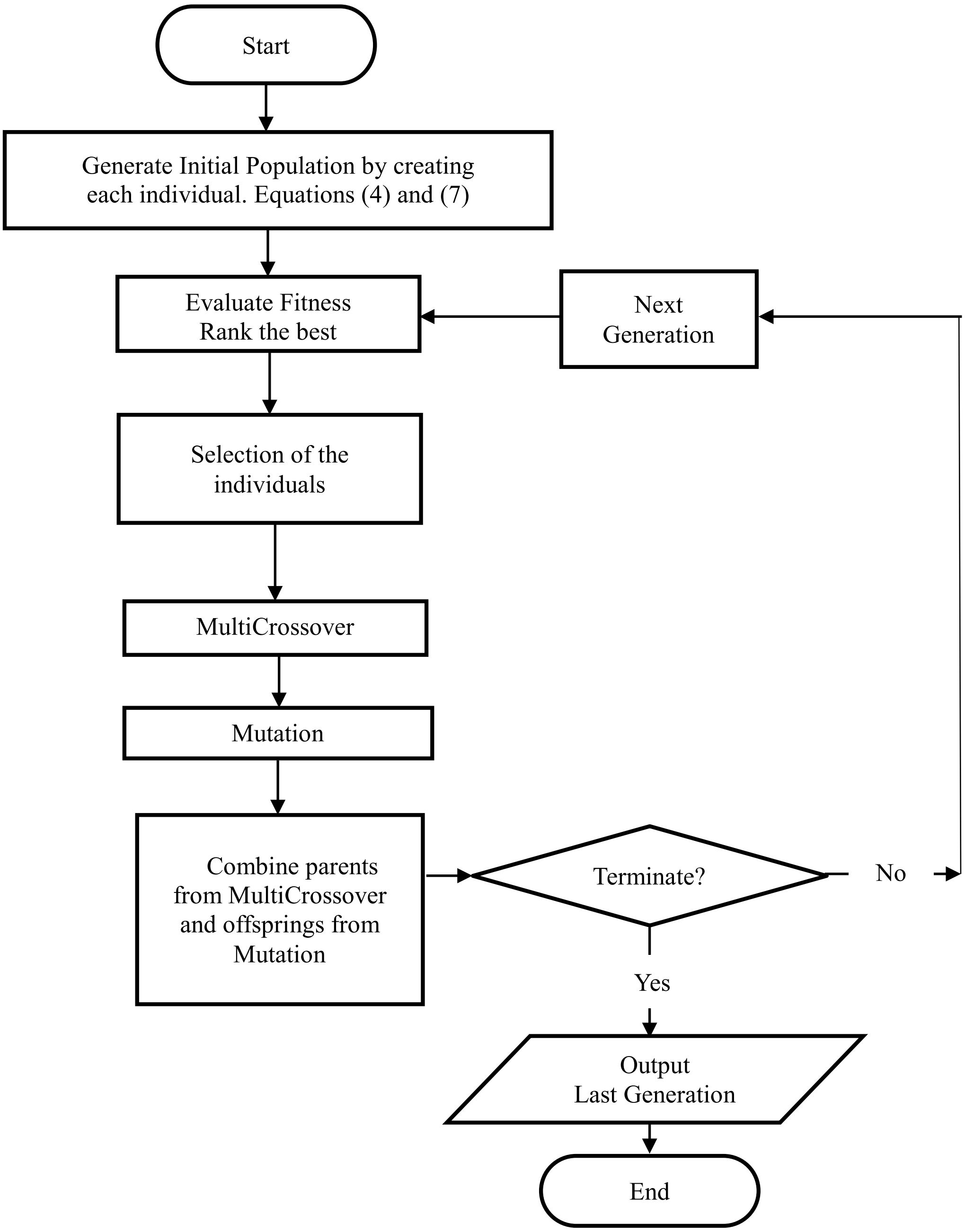


**Fig. 1.** The workflow of GA with Bound Constraint Strategy

constraint coefficients bound the search direction and the evolution process of a set of particles. The flow diagram of the modified PSO Algorithm is shown in Fig. 2.

The solution space among the particles relies on Lower and Upper bounds limits the number of combinations examined while looking for the solution. The optimal solution is a set of particles having minimum error and maximum profit. Each particle represents an item from the list, and the number of particles used is the length of the input items. The particle position in the swarm is initialized with 0; the initial swarm velocity is the random uniform value between the range 0 and $c \div \min(W)$. The size of the swarm is the length of the input items. Pbest, the best-known position of an individual and Gbest, the best position of the swarm are evaluated by the objective function, as shown in Equation (8) and Equation (9).

The evolution process of PSO is the computation of velocity and the direction of the particles in the swarm. The movement of particles is guided by various velocity rules, as discussed in Harman et al. (2015), to avoid the premature convergence to local minima. The modified PSO discussed in this paper, updates the particle's velocity and moves the

**Algorithm 1:** Bound strategy for Genetic Algorithm

```
Initialize Variables
gene ← βi #evaluate using Equation (7)
geneEncode ← maxItem
totalPopulation ← 100

InitialPopulation (geneLength, totalPopulation)
    iniPopulation ← Ø
    for i in range(0,totalPopulation) do
        iniPopulation ← ChromosomeGenerator(geneLength, totalItem)
Return iniPopulation

ChromosomeGenerator (geneLength, totalItem)
    Chromosome ← Ø
    lenChromosome ← r × totalItem
    for i in range(0, totalItem) do
        n ← random(geneLength[i])
        Chromosome ← decimalToBinary(n, geneEncode, lenChromosome)
Return Chromosome
```

particle-based on Bound Constraint coefficient $\gamma_i$ and $\beta_i$, respectively. Update the particle's velocity according to the relative values of Pbest and Gbest, using the following expression, as explained in the Algorithm 2.

The rate of change of velocity of the particles $V_{t+1}$ is computed using Equation (12).

$$V_{t+1} = \omega \times V_t + C_1 \times R_1 \times (Pbest - X_t) + C_2 \times R_2 \times (Gbest - X_t) \quad (12)$$

Where $C_1$ and $C_2$ are acceleration coefficients; $\omega$ is inertia weight; Gbest is global best position; Pbest is self best position. $R_1$, $R_2$ are modified with Bound Constraint coefficient $\gamma_i$ to control the magnitude of velocity to move towards Pbest and Gbest position with maximum profit.

Each particle's position is updated to accelerate them towards the best position found by it so far (Pbest) and the global best position (Gbest). The rate of change of position of the particles is computed using Equation (13).

$$X_{t+1} = X_t + V_{t+1} \quad (13)$$

In this way, the particle moves in search of the optimal solution until the maximum number of iterations is reached. The Algorithm 2 shows the implementation of a bounded strategy in updating the particles' position and velocity for achieving the optimal solution.

**Algorithm 2:** Bound strategy for PSO Algorithm

```
Initialize Variables
positionbounds← βi #evaluate using Equation (7)
velocitybounds← γi #evaluate using Equation (6)
num_dimensions←len(weight)

UpdateVelocity (pos_best_g, velocitybounds)
        for i in range(0,num_dimensions) do:
            r1[i] ← rand(velocitybounds)
            r2[i] ← rand(velocitybounds)
            Velocity[i] ← Vt+1
Return Velocity

UpdatePosition (positionbounds)
        for i in range(0,num_dimensions) do:
            Position[i] ← Xt+1
            if Position[i] > positionbounds[LBi]
                then Position[i] = positionbounds[LBi]
              if Position[i] < positionbounds[UBi]
                  then Position[i] = positionbounds[UBi]
Return Position
```

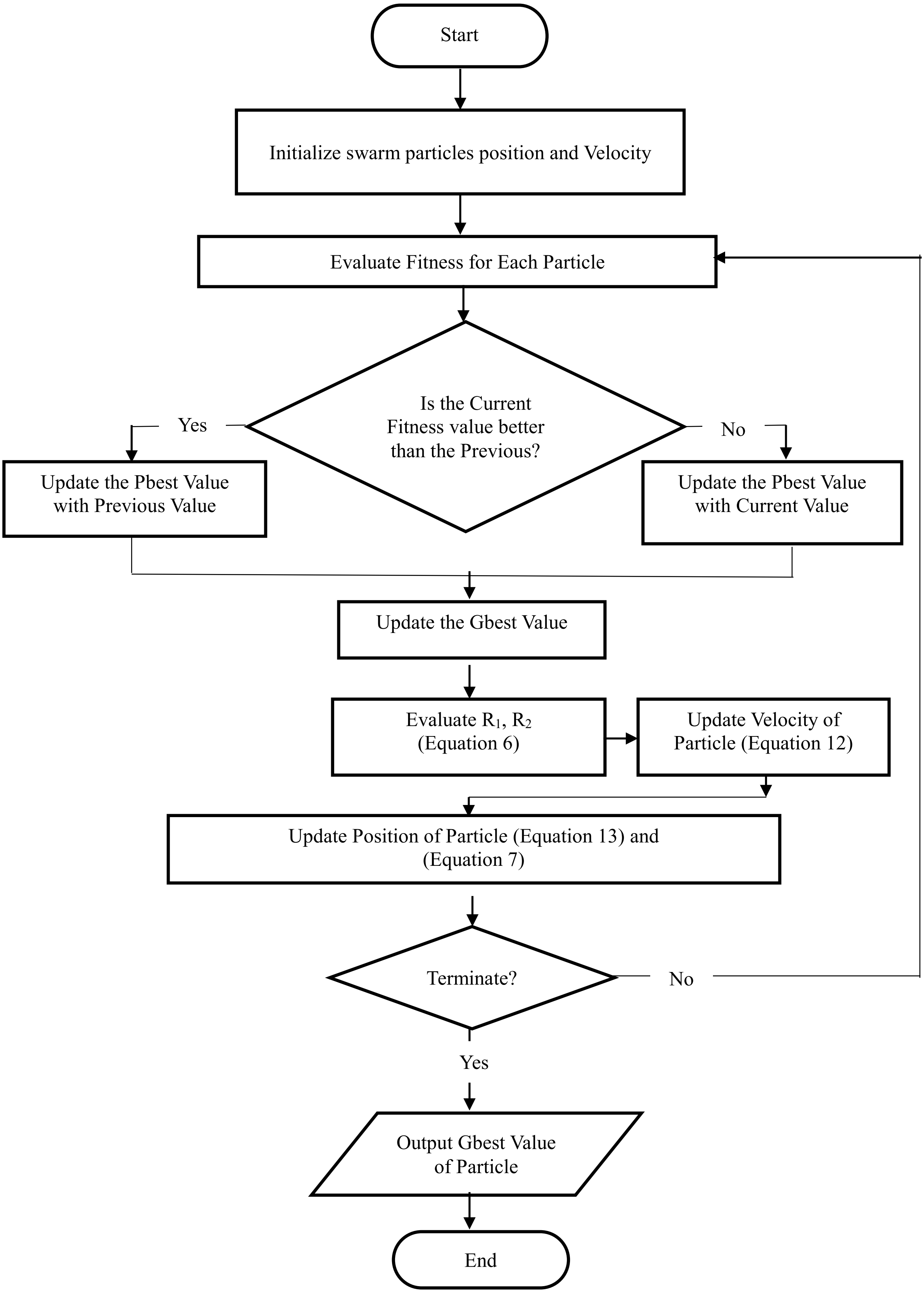


**Fig. 2.** The workflow of PSO with Bound Constraint Strategy

## 4. EXPERIMENTAL RESULTS AND DISCUSSION

### 4.1 Dataset

Considering different features of the optimization problem with different numbers and types of boundary constraints, as discussed in Michalewicz et al. (2000), scalable test data, as shown in Table 2, is generated for experimental purposes.

The optimum solution is found by varying the capacity($C$) = 150, 450 and 800.

The parameters used to design the Evolutionary Algorithms with bound constraints for the capacities 150, 450, and 800 are shown in Table 3. Some parameter values change with the capacity, and some remain constant. For example, the Mutation rate of GA and C1, C2, and ω of PSO remains constant, as shown in Table 3.

### 4.2 Simulation

The simulation engine is composed of modified GA and PSO models. The simulation engine is designed using the python platform. The set of initial conditions and parameters required by the simulation engine are defined in Table 3. The simulation engine creates an instance of the model in the simulation environment, applies the initial conditions to that instance, and then uses the equations expressed by the model to determine the change of state of that instance as a function of time. Random simulation for finding the optimal solution is computationally expensive, as discussed in Digabel and Wild (2015).

**Table 2.** Test Data for GA and PSO

| Item | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| weight($W$) | 70 | 73 | 77 | 80 | 90 | 94 | 98 | 106 | 110 |
| value($V$) | 135 | 139 | 149 | 150 | 173 | 184 | 192 | 201 | 210 |
| Item | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| weight($W$) | 113 | 115 | 118 | 120 | 122 | 125 | 130 | 135 | 140 |
| value($V$) | 214 | 221 | 229 | 240 | 245 | 250 | 253 | 255 | 258 |
| Item | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| weight($W$) | 142 | 145 | 148 | 152 | 154 | 155 | 156 | 158 | 160 |
| value($V$) | 262 | 268 | 272 | 275 | 280 | 282 | 285 | 288 | 292 |
| Item | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | |
| weight($W$) | 165 | 167 | 169 | 172 | 175 | 178 | 82 | 87 | |
| value($V$) | 295 | 300 | 302 | 308 | 412 | 415 | 156 | 163 | |

**Table 3.** Parameter settings for GA and PSO

| GA | | | PSO | | |
|---|---|---|---|---|---|
| Variable | Capacity($C$) | Values | Variable | Capacity($C$) | Values |
| Initial population size | 150 | 100 | Initial swarm particle | 150 | (0,0,…,0) |
| | 450 | 100 | Position | 450 | (0,0,…,0) |
| | 800 | 100 | | 800 | (0,0,…,0) |
| Genes | 150 | 2 | Swarm size | 150 | 35 |
| | 450 | 6 | | 450 | 35 |
| | 800 | 11 | | 800 | 35 |
| Chromosome length | 150 | 70 | C1(Constant) | - | 1 |
| | 450 | 210 | C2(Constant) | - | 2 |
| | 800 | 385 | ω (Constant) | - | 0.5 |
| Parent size | 150 | 25 | R1 | 150 | (0,2),…,(0,1) |
| | 450 | 25 | R1 | 450 | (0,2),…,(0,6) |
| | 800 | 25 | R1 | 800 | (0,4),…,(0,11) |
| Offsprings size | 150 | 25 | R2 | - | (0,1),…..,(0,3) |
| | 450 | 25 | $X_{t+1}$ | 150 | (0,2) |
| | 800 | 25 | $X_{t+1}$ | 450 | (0,6) |
| Multipoint crossover | 150 | 2 | $X_{t+1}$ | 800 | (0,11) |
| | 450 | 6 | Initial velocity | 150 | Random(0,2) |
| | 800 | 11 | Initial velocity | 450 | Random(0,6) |
| Mutation (Constant) | - | 0.5 | Initial velocity | 800 | Random(0,11) |
| Number of generation (Constant) | - | 50 | Number of iterations (Constant) | - | 7 |

The parameters used in the simulation are adaptive to improve the efficiency of the simulation engine.

The objective function and constraint coefficients defined in the modified EA are employed in each evolutionary mechanism for favorable convergence and diversity, as proposed in Sun et al. (2019). The total number of parents is (length of population ÷ 4), and the total number of offsprings is (length of population ÷ 2) − total number of parents. The parameter values are shown in Table 3.

### 4.3 Result

The converged optimal solution should have minimum error and maximum profit. Table 4 shows the optimal solution for GA with the capacity 150, 450 and 800 and the profit of 288, 864 and 1525, respectively. Table 4 shows that items were selected only once for capacity 150 and 450. So $I_s$ is a single copy of the item, whereas $I_s$ for capacity 800 is 7 copies of the item with weight 70 and 4 copies of the item with weight 77.

**Table 4.** Optimized solutions by GA

| Capacity(C) | Weight($W$) | Items Solution($I_s$) |
|---|---|---|
| 150 | 73 | 1 |
| | 77 | 1 |
| 450 | 70 | 1 |
| | 73 | 1 |
| | 77 | 1 |
| | 80 | 1 |
| 800 | 70 | 7 |
| | 77 | 4 |

Table 5 shows the optimal solution for PSO with a capacity of 150, 450 and 800 and a profit of 277, 850 and 1547. The optimal solution of PSO consists of only a portion of the items. For example, from Table 5. Item of weight 90 contributes 0.12 times, which is a fraction compared to the whole number in GA. But PSO and GA are designed for solving different industrial problems, a subset of the unbound knapsack problem. GA has the selection operator, but there is no selection operator in PSO. All individuals are kept as members of the population throughout the operation in PSO. However, PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity. They also have memory, which is essential to the algorithm.

The different GA and PSO applications are discussed: (1) PSO is used in budget scheduling problems. A portion of the budget can be used for different items. Finally, the total cost should not exceed the final allocated budget. (2) GA is used for Machine Scheduling Problems (MPS) to calculate the quantity required for enabling the efficient use of resources. Both these applications have the same objective to reduce the cost and contribute to a consistent rise in the company's profit margin. The reusability approach proposed in this paper is useful when two different departments of a company try to implement the cost reduction strategy. The PSO based model can be applied to the Finance Department, and the GA based model can be used to the Production Department.

**Table 5.** Optimized solutions by PSO

| Capacity(C) | Weight($W$) | Items Solution($I_s$) |
|---|---|---|
| 150 | 90 | 0.12 |
| | 110 | 0.13 |
| | 130 | 0.29 |
| | 142 | 0.10 |
| | 158 | 0.32 |
| 450 | 94 | 0.009 |
| | 106 | 0.03 |
| | 110 | 1.07 |
| | 120 | 1.15 |
| | 125 | 0.66 |
| | 140 | 0.05 |
| | 152 | 0.24 |
| | 154 | 0.24 |
| | 155 | 0.06 |
| | 156 | 0.16 |
| | 169 | 0.12 |
| 800 | 94 | 0.63 |
| | 110 | 0.05 |
| | 115 | 0.30 |
| | 122 | 0.16 |
| | 125 | 0.12 |
| | 130 | 1.03 |
| | 135 | 0.02 |
| | 148 | 0.60 |
| | 158 | 0.53 |
| | 169 | 0.21 |
| | 175 | 0.86 |
| | 178 | 0.92 |

To test the bounded strategy on the PSO Algorithm and the Genetic Algorithm, the objective function is plotted against iterations. The data seem to converge in max iteration ÷ 2 for PSO, as shown in Fig. 3a, Fig. 3c, and Fig. 3e. GA maintained the solution quality from the first generation to the last generation, as shown in Fig. 3b, Fig. 3d, and Fig. 3f. Our results show that the algorithms can be applied to different real-world scenarios.

In this experiment, the main objective is to find the minimum error and the maximum profit. But, if the decision-makers prefer to trade off with the error, then the objective function can be modified to get higher profit. For example, an optimized solution with the profit of 300 can be achieved if the acceptable error value is set between 0 and 10 as shown in Fig. 3a. Thus multiple optimized solutions are available based on the preference of the decision-makers.

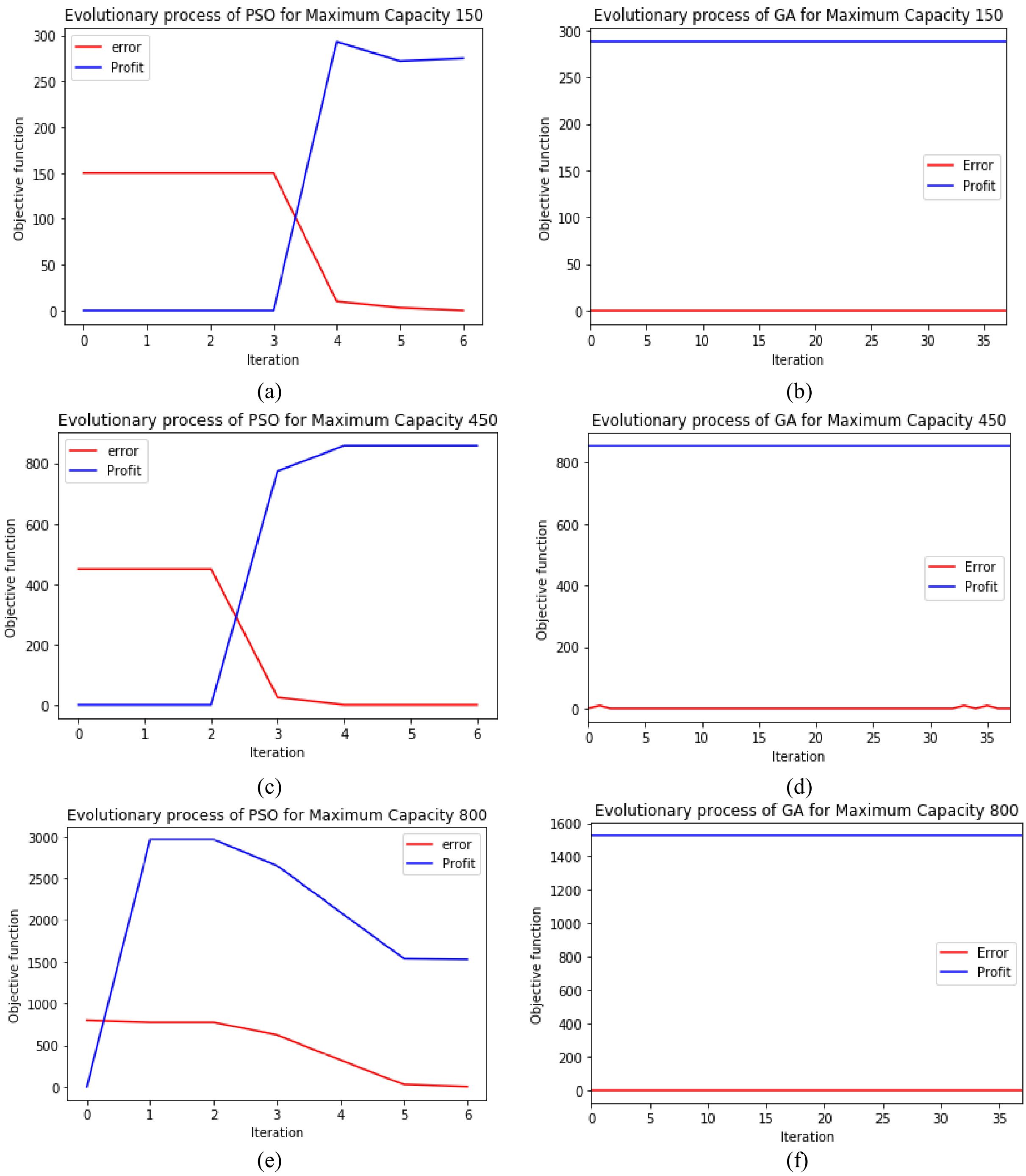


**Fig. 3.** Optimal solution convergence of PSO and GA

Execution time for PSO is better than GA in the benchmark algorithms discussed in Chaturvedi et al. (2016). Table 6 shows that the execution time of both the algorithms are the same if the same number of iterations are considered. The execution time cannot be kept as a benchmark for comparing the algorithms' performance since both the algorithms are applied to different types of UKP optimization.

**Table 6.** Execution time in seconds for GA and PSO

| Method | Capacity(C) | | | Number of Iterations |
|---|---|---|---|---|
| | 150 | 450 | 800 | |
| GA | 0.07133 | 0.08402 | 0.12416 | 50 |
| PSO | 0.01775 | 0.01823 | 0.01896 | 7 |

## 5. CONCLUSION

A modified EA with bound constraint strategy is present in this paper, to provide an optimized solution faster by constructing a feasible search space and a range of values bound constraints for the number of selection of an item. The bound constraints are determined by the minimum weight, maximum weight, minimum value, and maximum value of the items available in the search space to attain the optimal solution. The bound constraints coefficients are computed and implemented in the EA. The required strategy applied to EA modifies the GA and PSO, resulting in a different optimized solution for the same objective function. The summary of the series of the experiment results are as follows: (1) Solved UKP as a more general real-world problem whose optimum solutions were reached by bound constrain search strategies of EA. (2) The computational results demonstrated good performance and stability by varying the capacity and iterations. (3) The formulation of scalable constrained functions shows the algorithm's ability to deal with growing search space dimensions. (4) The reusability of the algorithm procedure in PSO and GA shows the EA's computational efficiency with a bound-constrained strategy. The proposed algorithm applies to UKP, but it can be extended to other similar problems, such as the Multiple Combination Problem, hard optimization problems, and bounded optimization problems. Part of the future work consists of designing hybrid methods that combine PSO and GA with backpropagation algorithms in finding the optimal weights and training the artificial neural networks for object detection technologies.

## ACKNOWLEDGEMENT

This paper is supported by the Ministry of Science and Technology, Taiwan. The No.s are MOST-107-2221-E-324 -018 -MY2 and MOST-106-2218-E-324 -002, Taiwan. This research is also partially sponsored by Chaoyang University of Technology (CYUT) and Higher Education Sprout Project, Ministry of Education (MOE), Taiwan, under the project name: "The R&D and the cultivation of talent for health-enhancement products."

## REFERENCES

Chai, R., Savvaris, A., Tsourdos, A., Chai, S., Xia, Y. 2020. Stochastic spacecraft trajectory optimization with the consideration of chance constraints, IEEE Trans. Control Syst. Technol., 28, 1550–1559, doi: 10.1109/TCST.2019.2908938.

Chaturvedi, S., Pragya, P., Verma, H.K. 2016. Comparative analysis of particle swarm optimization, genetic algorithm, and krill herd algorithm, IEEE Int. Conf. Comput. Commun. Control. IC4, 2015, 3–9, doi: 10.1109/IC4.2015.7375552.

Chen, X., Ong, Y.S., Lim, M.H., Tan, K.C. 2011. A multi-facet survey on memetic computation, IEEE Trans. Evol. Comput., 15, 591–607, doi: 10.1109/TEVC.2011.2132725.

Coello, C.C.A., Brambila, S.G., Gamboa, J.F., Tapia, M.G.C., Gómez, R.H. 2019. Evolutionary multiobjective optimization: open research areas and some challenges lying ahead, Complex Intell. Syst., 6, 221–236, doi: 10.1007/s40747-019-0113-4.

Corus, D., Oliveto, P.S. 2018. Standard steady state genetic algorithms can hillclimb faster than Mutation-Only Evolutionary algorithms, IEEE Trans. Evol. Comput., 22, 720–732, doi: 10.1109/TEVC.2017.2745715.

Deplano, I., Yazdani, D., Nguyen, T.T. 2019. The offline group seat reservation knapsack problem with profit on seats, IEEE Access, 7, 152358–152367, doi: 10.1109/ACCESS.2019.2948322.

Digabel, S.L., Wild, S.M. May, 2015. A taxonomy of constraints in Simulation-Based optimization, no. [Online]. Available: http://arxiv.org/abs/1505.07881.

Han, C., Zhang, P., Wang, W., Wang, Y., Zhang, Z. 2019. Delay-Optimal Joint processing in Computation-Constrained Fog radio access networks, IEEE Access, 7, 58857–58865, doi: 10.1109/ACCESS.2019.2913147.

Harman, E., Singh, P., Avneet Kaur, E. 2015. Statistical analysis of velocity update rules in particle swarm optimization, Int. J. Artif. Intell. Appl. Smart Devices, 3, 11–22, doi: 10.14257/ijaiasd.2015.3.2.02.

Hellwig, M., Beyer, H.G. 2019. Benchmarking evolutionary algorithms for single objective real-valued constrained optimization – A critical review, Swarm Evol. Comput., 44, 927–944, doi: 10.1016/j.swevo.2018.10.002.

Hsu, C.C., Yu, C.Y. 2004. Design of optimal controller for interval plant from signal energy point of view via evolutionary approaches, IEEE Trans. Syst. Man, Cybern. Part B Cybern., 34, 1609–1617, doi: 10.1109/TSMCB.2004.826396.

Jiang, H., Chen, R.C., Liu, Q.E., Huang, S.W. 2019. Fuzzy rules reduction based on sparse coding, International Journal of Applied Science and Engineering, 16, 215-227.

Juarez-Castillo, E., Acosta-Mesa, H.G., Mezura-Montes, E. 2017. Empirical study of bound constraint-handling methods in Particle Swarm Optimization for constrained search spaces, 2017 IEEE Congr. Evol. Comput. CEC 2017 - Proc., 604–611, doi: 10.1109/CEC.2017.7969366.

Li, S., Li, X., Wang, X., Liu, J. 2019. Usage-Constrained sensor selection, 65, 4392–4410.

Marius, O., Nicolae, P., Koprinkova, P., Yancho, T. 2017. Genetic algorithm for system modeling, Proc. 9th Int. Conf. Electron. Comput. Artif. Intell. ECAI, 2017-January, 1–4, doi: 10.1109/ECAI.2017.8166517.

Michalewicz, Z., Deb, K., Schmidt, M., Stidsen, T. 2000. Test-case generator for nonlinear continuous parameter optimization techniques, IEEE Trans. Evol. Comput., 4, 197–214, doi: 10.1109/4235.873232.

Nguyen, T.T., Yao, X. 2012. Continuous dynamic constrained optimization-the challenges, IEEE Trans. Evol. Comput., 16, 769–786, doi: 10.1109/TEVC.2011. 2180533.

Salomon, R. 1998. Evolutionary algorithms and gradient search: similarities and differences, IEEE Transactions on Evolutionary Computation, 2, 45–55, doi: 10.1109/ 4235.728207.

Satapathy, S.C., Naik, A., Parvathi, K. 2013. A teaching learning based optimization based on orthogonal design for solving global optimization problems, Springerplus, 2, 1–12, doi: 10.1186/2193-1801-2-130.

Sun, Q., Yang, G., Wang, X., Chen, Y.H. 2020. Regulating Constraint-Following bound for uncertain mechanical systems: An indirect control approach, IEEE Access, 8, 70193–70203, doi: 10.1109/ACCESS.2020.2981623.

Sun, Y., Yen, G.G., Yi, Z. 2019. IGD Indicator-Based Evolutionary algorithm for Many-Objective Optimization problems, 23, 173–187.

Yuan, B., Li, B., Chen, H., Yao, X. 2015. A new evolutionary algorithm with structure mutation for the maximum balanced biclique problem, IEEE Trans. Cybern., 45, 1040–1053, doi: 10.1109/TCYB.2014. 2343966.

Zhu, H., Alonso-Mora, J., 2019. Chance-Constrained Collision avoidance for MAVs in dynamic environments, IEEE Robot. Autom. Lett., 4, 776–783, doi: 10.1109/LRA.2019.2893494.