Benha Journal of Applied Sciences (BJAS) **print: ISSN 2356–9751**
Vol. (9) Issue (5) (2024), (289-320) **online: ISSN 2356–976x**
http://bjas.bu.edu.eg

Human-Based Optimization Algorithms and Their Applications

Mostafa.M.Saber[1], Mohamed.A.El Sayed[1, 2], Ahmed.N. Meshref [1], Mohamed.A.Elsisy[1]
1 Banha Faculty of Engineering, Banha University, Banha, EGYPT
2 Faculty of Engineering, BADR University in Cairo BUC, Cairo, Egypt
E-Mail: Prof.m.o.mounir@gmail.com

Abstract
A subset of meta-heuristic algorithms known as "human-based algorithms" are motivated by social interaction, human behavior, and problem-solving techniques. This study presents human-based meta-heuristic algorithms along with their benefits, drawbacks, and uses. This work provides an appraisal of the diversity of potential applications in optimization problems, the quick evolution of human-based meta-heuristic ideas, and their coverage towards a unified tissue. The purpose of the paper is to provide a quick overview of many human-based meta-heuristic methods for optimization problem-solving. Human-based optimization has at least eleven algorithms: The Driving Training-Based Optimization (DTBO), Chef-Based Optimization Algorithm (CBOA), Teaching-learning-based optimization (TLBO), Technical and Vocational Education, and Training-Based Optimizer (TVETBO), Sewing Training-Based Optimization (STBO), Volleyball Premier League Algorithm (VPL), Election-Based Optimization Algorithm (EBOA), Interior Search Algorithm (ISA), Social Engineering Optimizer (SEO), Human Behavior-Based Optimization (HBBO) and Seeker Optimization Algorithm (SOA). These algorithms mimic the problem-solving strategies humans employ to tackle complex optimization tasks. From simulated annealing to genetic algorithms, HBOAs encompass a diverse range of techniques, each offering unique advantages and applications across various domains.

Subjects: Meta-heuristics, Algorithms, Human-based algorithms, Mathematical models, Optimization Theory and Computation, Flow chart, Pseudo-Code, Advantages, Limitations, and Applications.

Keywords: Optimization, the problem of optimization, applied mathematics, population-based algorithms, stochastic algorithms, Population matrix.

Introduction

Meta-heuristic optimization methods fall into four main categories: swarm-based, human-based, physics-based, and phylogeny-based. A class of optimization algorithms known as "human-based algorithms" draws inspiration from social interaction, human behavior, and problem-solving techniques. These algorithms mimic the method by which people solve difficult problems; to arrive at a workable solution, they frequently draw on the experience and knowledge of several people.

There are various reasons why human-based meta-heuristic algorithms are necessary:

1. They are a simple instrument for reaching decision-making objectives.
2. Provide structure to the optimization problem so that it cannot be confirmed through mental calculation of the exact solution.
3. The definitions supplied with mathematical formulas fail to specify which alternatives must be established and how the problem's data must be gathered.
4. Can be used to help find the precise answer and for educational purposes.

However, there are drawbacks to human-based algorithms as well. For example, maintaining the consistency and quality of human input and dealing with concerns of fairness and bias. Notwithstanding these difficulties, it is anticipated that human-based algorithms will become more crucial in resolving complicated issues across a range of industries, including cyber security, healthcare, finance, and the best engineering applications by:

1. Accessible to combine with your existing implementation.
2. Not taking gradient information.
3. Can be applied to a wide range of problems including different topics.

Key Application Areas:

1. Engineering Design.
2. Healthcare.
3. Finance.
4. Supply Chain and Logistics.
5. Robotics and Automation.
6. Environmental Management.
7. Education Systems.

Human-based optimization algorithms are a subset of metaheuristic techniques that simulate human behavior, cooperation, learning, or competition to solve optimization problems. Unlike natural or physics-based methods, these algorithms explicitly mimic human strategies.

Human interactions and behavior have influenced the development of human-based algorithms. Of the algorithms in this group, Teaching-Learning-Based Optimization (TLBO) is the most popular and widely utilized. A key concept in the development of Doctor and Patients Optimization (DPO) has been the medical practitioner's approach to patient care [1], The Teamwork Optimization Algorithm (TOA) design was inspired by the interactions and cooperation of team members to carry out teamwork

and accomplish the intended aim [2], And few more human-based metaheuristic algorithms: Human Mental Search (HMS) [3], Poor and Rich Optimization (PRO) [4], Multi-Leader Optimizer (MLO) [5], Following Optimization Algorithm (FOA) [6], Tabu Search Algorithm [7].

Challenges

1. Balancing exploration and exploitation.
2. Handling high-dimensional data.
3. Ensuring scalability for real-world applications.

Future Directions

1. Hybridizing human-based algorithms with machine learning techniques.
2. Developing parallel implementations for faster computation.
3. Exploring more nuanced human behaviors like emotion and intuition.

The remainder of the paper is organized as follows: Section 2, presents a new human-based metaheuristic algorithm for solving optimization problems (DTBO) [8]. Section 3, shows a human-based metaheuristic optimization method based on mimicking cooking training (CBOA) [11]. Section 4, provides a Teaching-Learning-Based Optimization (TLBO) algorithm [15], [16]. In Section 5, submit Technical and Vocational Education and Training-Based Optimizer (TVETBO) [22]. Section 6, presents the human-inspired metaheuristic algorithm for solving optimization problems based on mimicking sewing training (STBO) [23]. Section 7, proposes a novel metaheuristic algorithm called Volleyball Premier League (VPL) [24]. Section 8, discusses a new optimization algorithm called the Election-Based Optimization Algorithm (EBOA) [29]. Section 9, Interior search algorithm (ISA): A novel approach for global optimization [34]. Section 10, illustrates The Social Engineering Optimizer (SEO) [40]. Section 11, introduces human behavior-based optimization (HBBO) [45]. Section 12, provides data about the seeker optimization algorithm (SOA) [50]. Finally, conclusions and several study proposals are presented in section 13.

2. DRIVING TRAINING-BASED OPTIMIZATION (DTBO) [8]

A relatively new metaheuristic optimization method called "Driving Training-Based Optimization" (DTBO) uses the driving training process as inspiration when addressing optimization issues.

The fact that driving training entails a mix of both exploration and exploitation forms the basis of the algorithm. In order to increase the student driver's experience and skill set, a driving instructor usually leads them on a variety of routes and scenarios. After a novice driver has accumulated some experience, the emphasis moves to making the most of that experience to refine their driving techniques and raise their level of performance.

This concept is turned into an optimization process in DTBO by combining guided search and random exploration. An initial population of randomly generated candidate solutions is the first thing the algorithm encounters. Then, using a series of driving training-inspired procedures, like "crossing over" and "mutation," the algorithm iteratively modifies this population in order to investigate new potential solutions.

Furthermore, the system integrates a human expert as a "driving instructor". The algorithm is guided by the expert's assessment of the potential solutions and comments on their quality. The quest for improved solutions is then further guided by the input received. The primary benefit of DTBO is its capacity to efficiently integrate exploration and exploitation, which may hasten the convergence of sound solutions. However, using a human expert also has a potential drawback because it necessitates having access to and knowledge of such an expert.

Taking everything into account, DTBO is a promising optimization technique that has proven effective on a number of benchmark projects. Further research is required to examine its potential and limitations in different issue domains.

DTBO's Mathematical Model

The members of DTBO, a population-based met heuristic, are instructors and learner drivers. Equation (1) uses a population matrix to simulate the given problem, and members of the DTBO are potential solutions to it. At the start of implementation, the placements of these members are initialized at random using Equation (2).

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{11} & \dots & x_{1j} & \dots & x_{1m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i1} & \dots & x_{ij} & \dots & x_{im} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N1} & \dots & x_{Nj} & \dots & x_{Nm} \end{bmatrix}_{N\times m} \quad (1)$$

$$x_{ij} = lb_j + r\,.\left(ub_j - lb_j\right), \quad i = (1,2,\dots,N, \quad j = 1,2,\dots,m) \quad (2)$$

where N is the size of the DTBO population, m is the number of problem variables, r is a random number from [0, 1], and lb_j and ub_j are the lower and upper bounds of the jth problem variable, respectively. X is the DTBO population, X_i is the *i*

th candidate solution, and x_{ij} is the value of the jth variable as determined by the i th candidate solution.

Each possible solution assigns values to the issue variables, which are then assessed for the objective function by incorporating them into the objective function. Consequently, the value of each possible quick fix for the objective function is computed. The vector in equation (3) represents the values of the objective function.

$$\mathrm{F} = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1} \qquad (3)$$

F is the vector of the objective functions, and F_i stands for the value of the objective function supplied by the i th candidate solution.

The values acquired for the objective function serve as the main yardstick for evaluating the quality of possible solutions. A comparison of the values of the objective function designates the best member X_{best} as the population member with the highest value for the objective function. Every time a candidate solution is refined and updated, the best member needs to be replaced as well.

The main difference between different heuristic approaches is the approach taken to update potential solutions. In DTBO, candidate solutions are updated in the following three stages: the student driver receives instruction from a driving teacher initially, then models the instructor's methods, and lastly, practices driving.

Phase1: Driving instructor training (exploration).

The DTBO update begins with the trainee driver selecting a driving instructor, who thereafter provides the novice driver with driving instruction. Once they have chosen the driving instructor and mastered their skills, members of the public will travel to several areas within the search area. Consequently, the DTBO will be able to search farther and discover the ideal spot with greater exploration capability. Thus, this DTBO update stage demonstrates the exploratory power of this method. Depending on how the data from each iteration compares The N members of the DTBO are selected as driving instructors by Eq. (4), taking into account the objective function.

$$\mathrm{DI} = \begin{bmatrix} DI_1 \\ \vdots \\ DI_i \\ \vdots \\ DI_{N_{DI}} \end{bmatrix}_{N_{DI}\times m} = \begin{bmatrix} DI_{11} & \cdots & DI_{1j} & \cdots & DI_{1m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ DI_{i1} & \cdots & DI_{ij} & \cdots & DI_{im} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ DI_{N_{DI}1} & \cdots & DI_{N_{DI}j} & \cdots & DI_{N_{DI}m} \end{bmatrix}_{\mathrm{N_{DI}\times m}} \qquad (4)$$

Where $N_{DI} = \left\lfloor 0.1 \times N \times (1 - \frac{t}{T}) \right\rfloor$ is the number of driving instructors, t is the current iteration and T is the maximum number of iterations, and DI is the matrix of driving instructors, DI_i is the ith driving instructor, and $DI_{i.j}$ is the j th dimension.

The mathematical modeling of this phase states that the new position for each member in this DTBO phase is initially established using Eq. (5). This new position replaces the previous one after Eq. (6) if it increases the value of the goal function.

$$x_{i,j}^{P_1} = \begin{cases} x_{i,j} + r\,.\,\left(DI_{k_{i,j}} - I\,.\,x_{i,j}\right), F_i > F_{DI_{k_i}}; \\ x_{i,j} + r\,.\,\left(DI_{k_{i,j}} - I\,.\,x_{i,j}\right), \quad otherwise, \end{cases} \qquad (5)$$

$$X_i = \begin{cases} X_i^{P_1}, F_i^{P_1} < F_i; \\ X_i, otherwise, \end{cases} \qquad (6)$$

Where $X_i^{P_1}$ is the new calculated status for the ith candidate solution based on the first phase of DTBO, $x_{i,j}^{P_1}$ is its jth dimension, Its objective function value is $F_i^{P_1}$, its j th dimension is $DI_{k_{i,j}}$, its ith member is trained by a randomly selected driving instructor, k_i is randomly selected from the set {1, 2,...,}, and I is a randomly selected number from {1, 2}, r is a randomly selected number in the interval [0, 1], and its objective function value is $F_{DI_{k_i}}$.

Phase 2: The student driver's instructor skills are patterned through experimentation .

In the second level of the DTBO update, the trainee driver attempts to mimic every gesture and driving style used by the instructor. To increase the DTBO's capacity for exploration, its members are dispersed around the search space. To quantitatively simulate this idea, a new position is formed based on the linear combination of each member with the teacher, as per Eq. (7). If the new position raises the value of the objective function, then Eq (8) states that it will replace the old one.

$$\mathrm{x_{i,j}^{P_2}} = \mathrm{P}\,.\,\mathrm{x_{i,j}} + (1 - \mathrm{P})\,.\,\mathrm{DI_{k_{i,j}}}\,, \qquad (7)$$

$$X_i = \begin{cases} X_i^{P_2}, & F_i^{P_2} < F_i; \\ X_i, & \text{otherwise}, \end{cases} \quad (8)$$

Where $X_i^{P_2}$ is the new calculated status for the ith candidate solution based on the second phase of DTBO, $x_{i,j}^{P_2}$ is its jth dimension, is $F_i^{P_2}$ the value of its goal function, and P is the patterning index provided by

$$P = 0.9\left(1 - \frac{t}{T}\right) + 0.01 \quad (9)$$

Phase 3: Individual application (exploitation) .

The third iteration of the DTBO update is focused on helping student drivers hone and enhance their driving skills. At this point, every beginner driver wants to get just a bit closer to his peak performance. During this phase, each participant is permitted to conduct a local search in the vicinity of their current location in order to find a better position. This phase shows how DTBO can make use of local search. Through mathematical modeling, this DTBO phase is modeled such that an initial random position is established near each member of the population by Eq. (10). The previous position is then replaced using Eq. (11) if this position raises the value of the goal function.

$$x_{i,j}^{P_3} = x_{i,j} + (1 - 2r) \cdot R \cdot \left(1 - \frac{t}{T}\right) . \quad (10)$$

$$X_i = \begin{cases} X_i^{P_3}, & F_i^{P_3} < F_i; \\ X_i, & \text{otherwise}, \end{cases} \quad (11)$$

Where $X_i^{P_3}$ is the new calculated status for the *i* th candidate solution based on the third phase of DTBO, $x_{i,j}^{P_3}$ is its jth dimension, T is the maximum number of iterations, R is the constant set to 0.05, r is a random real number in the interval [0, 1], and $F_i^{P_3}$ is the value of its Objective function.

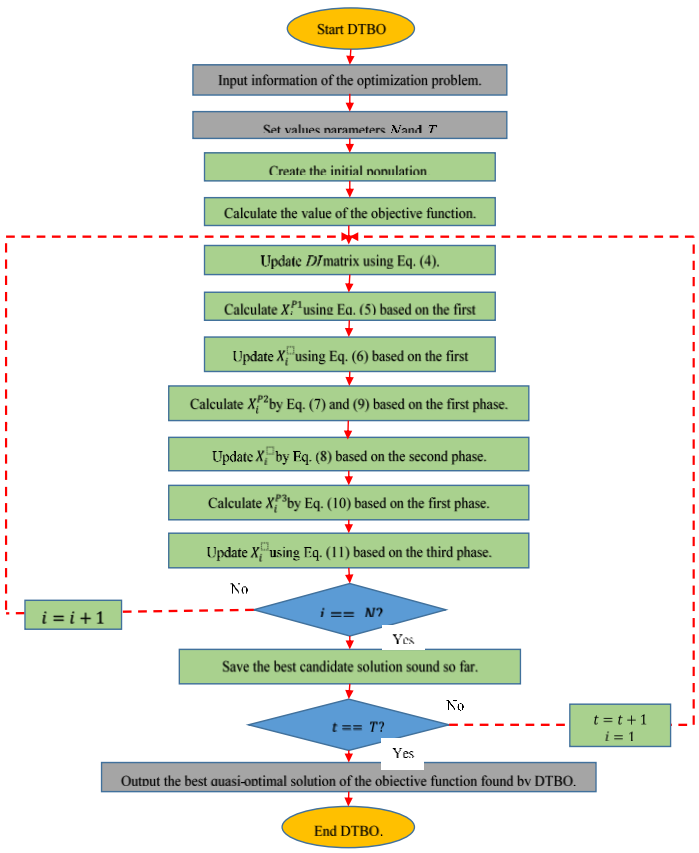
DTBO Pseudo-Code # 1. [8]

Start DTBO.

1. Input: The details of the optimization problem.
2. Modify N and T.
3. Set the DTBO population position initial and assess the goal function.
4. For t = 1 to T
5. For i = 1 to N
6. Phase 1: Driving instructor training
7. Comparing the values of the objective function will allow you to determine the driving instructor matrix.
8. From the matrix DI, choose a driving instructor
9. Use Eq. (5), get the *i* th DTBO member's new.
10. Eq. (6) gets the *i* th DTBO member's position.
11. Phase 2: The student driver's instructor skills.
12. Use Eq. (9), to determine the patterning index P.
13. Use Eq. (7), get the *i* th DTBO member's new.
14. Eq. (8) gets the *i* th DTBO member's position.
15. Phase 3: Individual application (exploitation).
16. Use Eq. (10), get the *i* th DTBO member's new.
17. Eq. (11) get the *i* th DTBO member's position.
18. End.
19. Update the solution found to be the best.
20. End.
21. Output: The optimal potential solution.

End DTBO .

Figure (1): The DTBO flow chart. [8]



The benefits of DTBO include:

1- The roles the optimization findings for C1 through C30 showed that DTBO can successfully handle difficult optimization problems.

2- When the suggested DTBO's performance was compared to that of rival algorithms, it was shown to be substantially more competitive and effective in optimizing and achieving optimal solutions than the algorithms assessed.

3- Using DTBO to solve two engineering design difficulties demonstrated how well the recommended approach worked to resolve real-world issues.

DTBO has certain limitations:

1- Although DTBO has shown promise in solving problems, there are certain limitations with this method in other contexts .

2-The NFL theorem categorically and categorically refutes the authors' claim that DTBO is the best optimizer for handling optimization problems. As a result, the authors make no such claims. Because of this, DTBO may not be able to solve all optimization challenges .

3-The main disadvantage of any metaheuristic algorithm, including DTBO, is that it's possible that in the future, new optimization methods will be developed to handle optimization applications more skillfully.

DTBO applications:

1) The real-world optimization goal of pressure vessel design is to minimize design costs. [9]

2) Design of welded beams: Reducing manufacturing costs is the aim of this engineering optimization issue. [10]

3. COOKING-BASED OPTIMIZATION ALGORITHM (CBOA) [11]

A new optimization technique called the Cooking-Based Optimization Algorithm (CBOA) is inspired by the cooking process and culinary education. This method seeks to solve optimization problems by emulating the way chefs mix materials and employ various cooking methods to produce a delectable dish.

CBOA is a population-based meta-heuristic optimization algorithm that makes use of a set of "dishes," or potential solutions, that have been developed via the use of different cooking methods like boiling, frying, and baking. The foundation of the algorithm is the idea that the best solutions are those that employ appropriate cooking methods and ingredient combinations in the proper ratios to produce the desired results.

First, a randomly created starting set of dishes is used by the CBOA algorithm. Every dish is assessed using a fitness function that gauges how well it meets the requirements of the optimization task at hand. The dishes are ranked and chosen for additional preparation in accordance with this assessment.

After that, the CBOA algorithm cooks the chosen dishes using a variety of methods, including crossover, mutation, and selection, to produce new dishes that might be superior to the ones that came before them. Until a workable solution is identified or a predefined stopping requirement is satisfied, this process is repeated.

The ability of CBOA to solve intricate optimization problems with plenty of variables and constraints is one of its advantages. Its ability to quickly and effectively incorporate restrictions and domain-specific knowledge into the optimization process is another benefit.

But CBOA has drawbacks and difficulties just like any other optimization algorithm. The performance of the algorithm can be greatly impacted by the choice of suitable cooking methods and their parameters, which presents one issue. An additional difficulty is the possibility of early convergence or becoming trapped in local optima, which might lessen the algorithm's efficiency.

In summary, a promising new optimization strategy that leverages culinary methods to solve challenging optimization problems is the Cooking-Based Optimization Algorithm (CBOA).

Even though it has drawbacks and difficulties, it could be a useful tool in a variety of fields, including engineering, finance, and logistics.

CBOA mathematical modelling:

If the rows of the CBOA population matrix are sorted in ascending order according to the value of the objective function (i.e., the member in the first row is the best member), then the group of the first N_C members is chosen as the group of chef instructors, and the remaining group of N - N_C members is chosen as the group of cooking students. Eqs. (12) and (13), respectively, describe the sorted objective function vector and the CBOA sorted population matrix.

$$\mathrm{XS} = \begin{bmatrix} XS_1 \\ \vdots \\ XS_{N_C} \\ XS_{N_C+1} \\ \vdots \\ XS_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ xs_{N_C,1} & & xs_{N_C,j} & & xs_{N_C,m} \\ xs_{N_C+1,1} & \cdots & xs_{N_C+1,j} & \cdots & xs_{N_C+1,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ xs_{N,1} & \cdots & xs_{N,j} & \cdots & xs_{N,m} \end{bmatrix}_{N\times m}, \quad (12)$$

$$\text{FS} = \begin{bmatrix} FS_1 \\ \vdots \\ FS_{N_C} \\ FS_{N_C+1} \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} , \quad (13)$$

Where the vector of ascending objective function values is called FS, the sorted population matrix of CBOA is called XS, and the number of chef instructors is denoted by N_C. The group of chef instructors is represented by members from XS_1 to XS_{N_C} in the matrix XS, whereas the group of cooking students is represented by members from XS_{N_C+1} to XS_N. The values of the objective functions corresponding to XS_1 to XS_N are successively included in the vector FS_1.

Phase 1 involves updating XS_1 to XS_{N_C} for a group of chef instructors.

At a culinary school, it is generally accepted that several chef teachers are in charge of teaching students cooking techniques. Two techniques are used by chef educators to improve their culinary skills. They attempt to mimic the top culinary instructor and comprehend their techniques in the first tactic. This strategy demonstrates the power of global search and CBOA investigation.

The advantage of upgrading the chef instructors using this method is that, before starting to train students, the best chefs (i.e., the top population members) enhance their skills depending on them. Because of this, the CBOA design does not directly depend on elevating a student's standing based just on the best members of the population.

This tactic also increases the precision and efficiency with which various search space regions are analyzed and keeps the algorithm from becoming trapped in local optima. Using the following equation, a new location for each chef teacher is initially determined based on this technique for i = 1, 2... N_C and j = 1, 2... m.

$$xs_{i,j}^{c/s_1} = xs_{i,j} + r\,.\left(BC_j - I.xs_{i,j}\right), \quad (14)$$

Where, based on the first strategy c/s_1 of updating the chef instructor, $xs_{i,j}^{c/s_1}$ is the new calculated status for the i th sorted member of CBOA (that is xs_i). The j th coordinate of this is (c/s_1); BC is the best chef instructor (represented as xs_1 in the matrix xs); BC_j is the best chef instructor's j th coordinate; r is a random value from the range [0, 1], and I is a number that is randomly selected during execution from the set {1, 2}. The CBOA will accept this revised position as long as it raises the goal function's value. Equation (15) is used to model this state.

$$XS_i = \begin{cases} XS_i^{c/s_1}, & FS_i^{c/s_1} < F_i \ ; \\ XS_i, & otherwise \ , \end{cases} \quad (15)$$

where FS_i^{c/s_1} is the member XS_i^{c/s_1} 's objective function value.

In the second method, each chef instructor uses customized workouts and activities to hone his cooking talents. This method shows how the CBOA can be used for both local search and exploitation. A chef teacher will try to improve each problem variable if they see them as individual cooking talents, with the goal of increasing the objective function value.

The advantage of updating based on personal actions and workouts is that, regardless of the location of other population members, each member looks for better options nearby. Better answers can be found using local search and exploitation by making little changes to the population members' locations in the search space. A random position is generated around every culinary instructor in the search region for j = 1, 2...., m, using Equations (16) to (17). Equation (11) is used to describe this condition, which says that this random position should be updated if it raises the objective function's value.

$$lb_j^{local} = \frac{lb_j}{t} \ , \quad (16)$$

$$ub_j^{local} = \frac{ub_j}{t} \ , \quad (17)$$

The variable t denotes the iteration counter, and the lower and upper local bounds of the j th problem variable are represented by the expressions lb_j^{local} and ub_j^{local}, respectively.

$$xs_{i,j}^{c/s_2} = xs_{i,j} + lb_j^{local} + r\,.\left(ub_j^{local} - lb_j^{local}\right), i = 1,2,\dots,N_C, j = 1,2,\dots,m \quad (18)$$

$$XS_i = \begin{cases} XS_i^{C/S_1}, & FS_i^{C/S_2} < F_i \ ; \\ XS_i, & otherwise \ , \end{cases} \quad (19)$$

where $xs_{i,j}^{c/s_2}$ is its j th coordinate, FS_i^{C/S_2} is its value of the objective function, and XS_i is the new computed status for the i th CBOA sorted member (i.e.,XS_i) based on the second strategy (c/s_2) of chef instructors updating.

Phase 2: the cookery students' group updating procedure (from XS_{N_C+1} to XS_N).

Students who wish to become chefs or learn how to cook enroll in culinary schools. Three assumptions about how cooking students would learn are built into the CBOA design. As per the initial plan, every student interested in culinary arts chooses a class at random from which a chef trains him on cooking methods. This approach has the advantage of providing cooking students with guidance from a variety of chef instructors. As a result, depending on the guidance of the chosen chef instructor, cooking pupils pick up different talents (i.e.,

population members travel to different locations of the search space). On the other hand, if every culinary student only received instruction from the greatest chef-instructor that is, if every member of the population gravitated towards the best then a successful worldwide search in the field of problem-solving would be impossible.

The CBOA simulates this technique by first utilizing Eq. (20) to calculate a new position for each cooking student based on their training and instruction from the chef instructor for i=$N_C + 1, N_C + 2$,…,N, j=1,2,…,m.

$$xs_{i,j}^{S/S_1} = xs_{i,j} + r \cdot \left(CI_{k_{i,j}} - I \cdot xs_{i,j}\right), \quad (20)$$

The new calculated status for the i th sorted member of CBOA (i.e., XS_i) is represented by XS_i^{S/S_1}, which is based on the first strategy (S/S_1) of the cooking students updating; its j th coordinate is $xs_{i,j}^{S/S_1}$ and the selected chef instructor by the i th cooking student is represented by $CI_{k_{i,j}}$ where K_i is randomly selected from the set {1,2,..., N_C} (where $CI_{k_{i,j}}$ denotes the value $xs_{k_{i,j}}$.

If the new position increases the value of the goal function, it takes the place of the prior one for each member of the CBOA. Equation (21) models this idea for i=$N_C + 1, N_C + 2, \dots N$

$$XS_i = \begin{cases} XS_i^{S/S_1}, & FS_i^{S/S_1} < F_i \ ; \\ XS_i, & otherwise \ , \end{cases} \quad (21)$$

where the objective function value of XS_i^{S/S_1} is represented by FS_i^{S/S_1}.

This approach enhances the CBOA's capability to seek and explore the globe. This method has the advantage of having just one variable in this example, one talent. It might not be necessary to update every member's location coordinate in order to get better answers. This "skill" in the CBOA design denotes a particular element of a vector of cooking skills of a chef instructor CI_k (k ∈ {1, 2... N_C}) chosen at random. Therefore, the second strategy is simulated mathematically in such a way that, for each cooking student XS_i (a member of CBOA with i=$N_C + 1, N_C + 2, \dots N$, first one chief instructor is randomly selected (a member of CBOA with the index K_i which is randomly selected from the set {1,..., N_C}), represented by the vector CI_{k_i} = ($CI_{k_{i,1}}$,…., $CI_{k_{i,m}}$), Then, we replace the ↕ th coordinate of the vector of the i th cooking student XS_i (thus, $XS_{i,↕}$) with this value $CI_{k_{i,↕}}$ by randomly selecting his ↕ th coordinate (thus, a number ↕ from the set {1,...m}, which reflects a "skill" of this picked chief teacher).

In accordance with this idea, Eq. (22) is used to compute a new position for every CBOA cooking student member.

$$xs_{i,j}^{S/S_2} = \begin{cases} CI_{k_{i,j}}, & j = l \ ; \\ XS_{ij}, & otherwise \ , \end{cases} \quad (22)$$

Where $i = N_C + 1, N_C + 2, \dots, N, j = 1,2, \dots, m$, and ↕ is a randomly chosen number from the range {1, 2... m}.

If it raises the goal function's target value, the previous location based on Eq. (23), is substituted.

$$XS_i = \begin{cases} XS_i^{S/S_2}, & FS_i^{S/S_2} < F_i \ ; \\ XS_i, & otherwise \ , \end{cases} \quad (23)$$

Where XS_i^{S/S_2} is the new calculated status for the i th sorted member of CBOA (i.e.,XS_i) based on the second strategy (S/S2) of updating cooking students, $xs_{i,j}^{S/S_2}$ is its j th coordinate, FS_i^{S/S_2} is its objective function value.

The third strategy has each student using exercises and activities to refine their cooking skills. This method shows how the CBOA can be used for both local search and exploitation. If the problem variables are viewed as cooking talents, then a cooking student will aim to improve each one to increase the objective function value.

According to this theory, Eqs. (16) and (17) provide a random position around each culinary student in the search space, and Eq. (24) derives a new position.

$$XS_i^{S/S_3} = \begin{cases} xs_{i,j} + lb_j^{local} + r \cdot \left(ub_j^{local} - lb_j^{local}\right), & j = q; \\ xs_{i,j}, & j \neq q \ , \end{cases} \quad (24)$$

where XS_i^{S/S_3} is its j th coordinate, and q is a randomly chosen number from the set {1, 2,..., m}, i=$N_C + 1, N_C + 2$,…, N, j=1,2,…,m. Based on the third strategy (S/S_3) of updating cooking students, XS_i^{S/S_3} is the new calculated status for the i th sorted member of CBOA (that is, xs_i). This new random position is suitable for updating XS_i, which is modeled by Eq. (25) if it increases the value of the goal function.

$$XS_i = \begin{cases} XS_i^{S/S_3}, & FS_i^{S/S_3} < F_i \ ; \\ XS_i, & otherwise \ , \end{cases} \quad (25)$$

CBOA Pseudo-Code #2. [11]

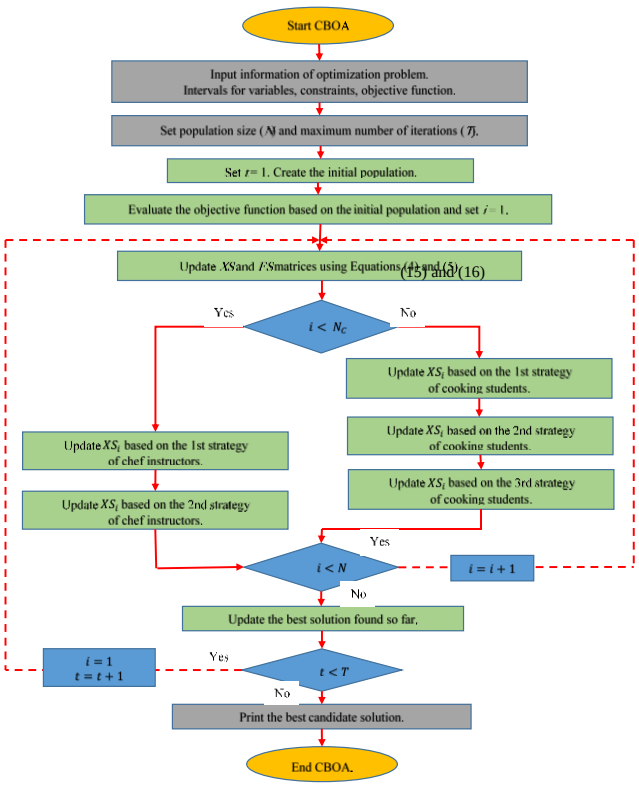
Begin CBOA.

1. Variables, objective function, and restrictions
2. Get (T) and CBOA population (N).
3. Create an initial population matrix X.
4. Obtain the vector F provided objective
5. For t = 1 to T
6. Use Eq. (12) and (13), to sort the matrix X according to the values.

7. Update the group of chef instructors CI = $\{CI_1, CI_2, \ldots, CI_{N_C}\}$ along with the top CBOA member BC.
(To be clear, we set $= CI_1$)
8. Phase One: Driving teacher training begins.
9. For i = 1 to N_C
10. Determine XS_i^{S/S_1} by utilizing Eq.(14)
11. Equation (15) is used to update XS_i.
12. Eq. (16) and (17) use to variables' upper and lower local bounds.
13. Determine XS_i^{S/S_2} by applying Eq.(18)
14. Use Equation (19) to update XS_i.
15. End.
16. End Phase 1: updating (XS_{1}, XS_{N_C})
17. Start Phase 2: updates (XS_{N_C+1}, XS_{N})
18. For $i = N_C + 1$ to N.
19. The i th cooking student will be trained by a chef instructor chosen at random.
20. Get XS_i^{S/S_1} by using Eq. (20).
21. Equation (21) is used to update XS_i.
22. Use Eq. (22) compute XS_i^{S/S_2}
23. Equation (23) is used to update XS_i.
24. Use Eq. (16), (17), and (24) get XS_i^{S/S_3}
25. Equation (25) is used to update XS_i.
26. End.
27. End Phase 2: updated (XS_{N_C+1}, XS_{N}).
28. Get the currently best candidate solution
29. End.
30. Output: The optimal resolution

End CBOA .

Figure (2): The flow chart for CBOA. [11]



The benefits of CBOA include:

1. The implementation outcomes of CBOA and rival algorithms on functions F8 through F13 show how well CBOA may be used for global search exploration across a range of problem-solving areas.
2- The optimization results for functions F9 and F11 demonstrate this CBOA potential's extraordinary strength.
3. It can utilize global search to find the primary optimal region before using local search to converge to the global optimum because of its ability to strike a balance between exploitation and exploration.
4- The CBOA outlines the difficulties and paths for upcoming study.
5- Using CBOA for optimization applications in many disciplines and real-world issues is another recommendation.
6- The optimization problem-solving ability of CBOA is tested and compared against fifty-two common benchmark functions and twelve popular meta-heuristic methods.

CBOA has certain limitations:

1- There are no specific requirements for its effective operation in any optimization application. Because of this, there is a disadvantage and a restriction.
2- Scholars may always develop more advanced that provide better solutions to real-world optimization problems than those made possible by already-existing algorithms.
3- Many competing algorithms, however faster, were unable to produce the desired results. Because of this, CBOA has a reasonable execution time while optimizing the goal functions.

CBOA applications:

1-Pressure vessel design (PVD). [9]
2- Welded beam design (WBD). [10]
3-Structural tension/ compression springs (TCSD). [12]
4-Speed reducer design (SRD). [13], [14]

TEACHING–LEARNING-BASED OPTIMIZATION (TLBO) [15], [16]

The Teaching-Learning-Based Optimization (TLBO) algorithm, as described, is inspired by the dynamics between a teacher and students in a classroom setting. The algorithm has two distinct phases: the "teacher phase" and the "learner phase."
Teacher Phase: In this phase, the algorithm emulates the concept of learning from a teacher. The teacher is metaphorically represented as a highly educated entity, symbolizing knowledge and expertise.
The quality of the teacher's influence is crucial, as it is believed to impact the overall performance or outcome of the learning process.

Analogous to a good teacher contributing to better student grades, the algorithm aims to leverage a "teacher" component to guide the optimization process.

Learner Phase: Following the teacher phase, the algorithm enters the learner phase, where learning occurs through interaction among learners.

This phase involves collaborative learning or the exchange of information between different components or entities within the algorithm.

The emphasis is on interaction and mutual learning, mirroring the way students in a class might learn from each other through discussions and shared experiences.

The TLBO algorithm essentially uses this teacher-learner paradigm to optimize a solution or search space. The "teacher" imparts information or guidance based on their knowledge, and the "learners" interact and learn from each other, fostering a collaborative optimization process.

The success of the TLBO algorithm is contingent upon the effectiveness of both the teacher and learner phases, with the algorithm aiming to strike a balance between exploration and exploitation in the optimization landscape. It is a heuristic optimization algorithm that draws inspiration from the dynamics of a classroom to address optimization problems in various domains.

1- Teacher phase

In this phase, the best member of the community is selected as the teacher or instructor and directs the average population toward himself. This is similar to what a real-world teacher does. This step is formulated as follows:

$$X_{new}^{k} = X_{old}^{k} + r\left(X^{teacher} - T_F * M(j)\right) \quad (26)$$

$$M(j) = \frac{\sum_{K=1}^{N} \frac{X^K(j)}{F^K}}{\sum_{K=1}^{N} \frac{1}{F^K}} \quad (27)$$

Where $X^K(j)$ represents the j-th design variable, T_F is used as the training factor, r is a random number in the range [0, 1], $M(j)$ indicates the average of the class, and F^K denotes the 0penalty fitness function.

2- Learning phase

In this phase, the people in the population (who are classmates) develop their knowledge by working together. This is similar to what happens to friends and classmates. This step is formulated as follows: Students p and q are randomly selected from the class so that they are unequal, thus:

If $X_p < X_q$

$$X_{new}^{p} = X_{old}^{p} + r\,(X_{old}^{p}(j) - X^{q}(j)) \quad (28)$$

Otherwise

$$X_{new}^{p} = X_{old}^{p} + r(X^{q}(j) - X_{old}^{p}(j)) \quad (29)$$

where r is a random number in the range [0, 1], and $X^P(j)$ represents the j-th design for the *p*-th design vector.

Pseudo-Code of TLBO #3.

Begin TLBO.

1. Enter the variables, an objective function,.
2. Phase 1: Teacher.

teacher Phase ($X^{teacher}$ population, TF, r):

best _teacher = Best (X _population)

average e_ population = calculate Average

for each individual $X_{\square}^{k}$ in X _population:

M (j) = calculate Average Design Variable (average_ population, X_ population, TF)

random_ number = generate Random Number in Range (0, 1)

for each design variable j in X^K:

$X_{new}^{k} = X_{old}^{k} + r\left(X^{teacher} - T_F * M(j)\right)$

return X_ population

3. Phase 2: Learning.

function learning Phase (X_ population, r):

for each pair of students p, q (p! = q):

If $X_p < X_q$

for each design variable j:

$X_{new}^{p} = X_{old}^{p} + r\,(X_{old}^{p}(j) - X^{q}(j))$

else:

for each design variable j:

$X_{new}^{p} = X_{old}^{p} + r(X^{q}(j) - X_{old}^{p}(j))$

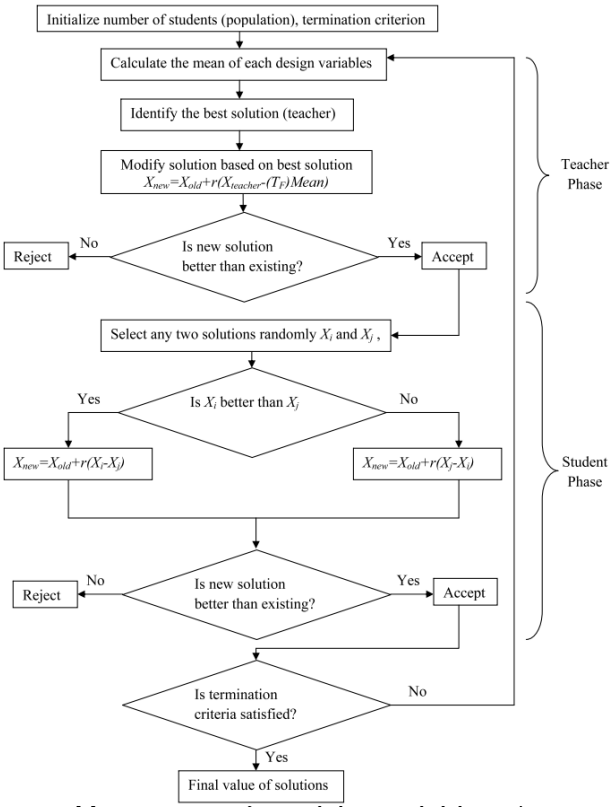
return X_ population

End

Output: The optimal quasi-optimal resolution

End TLBO .

Figure (3): The flow chart for TLBO. [15]



The benefits of TLBO include:

1- Intuitive Conceptual Basis: Inspired by teaching and learning processes, making it easy to understand.
2- Simple Implementation: Straightforward algorithmic structure, easy to implement.
3- Efficiency: Competitive performance, quick convergence to near-optimal solutions.
4- Population Diversity: Maintains diversity, and explores different regions of the search space.
5- Versatility: Applicable to various optimization problems (continues/discrete, linear/nonlinear).
6- Fewer Tunable Parameters: Requires fewer parameters to be tuned, simplifying its usage.
7- Robustness: Performs well across different problem types and dimensions.
8- Parallelization Potential: Easily parallelizable, suitable for parallel and distributed computing.
9- Scalability: Adaptable to problems of different sizes and complexities.

The limitations of TLBO are:

1- Sensitivity to population initialization
2- Convergence speed concerns
3- Parameter sensitivity
4- Challenges in handling constraints
5- Vulnerability to local optima
6- Difficulty in fine-tuning
7- Limited theoretical understanding
8- Limited performance comparison against other algorithms

Applications of TLBO:

1- Different manufacturing fields such as Milling, Drilling, Turning, Grinding, Electric Discharge Machining, Abrasive Jet Machining, Ultrasonic Machining, Electrochemical Machining, Laser Beam Machining, Micro Machining, etc. [17]
2- Solve multi-dimensional, linear and nonlinear problems with appreciable efficiency.[18]
3- For the discrete optimization of truss structures. [19].
4- Application to synthesis of sparse concentric ring arrays. [20]
5- Solving the optimal power flow problem with stochastic wind and so power generators [21].

5. Technical and Vocational Education and Training-Based Optimizer (TVETBO) [22]

The process of teaching work-related skills at technical and vocational education and training institutions served as the inspiration for a new metaheuristic algorithm known as the Technical and Vocational Education and Training-Based Optimizer (TVETBO). The three stages of the mathematical modeling of the algorithm—theory education, practical education, and individual skill development—are intended to address optimization problems. The evaluation of TVETBO's performance is conducted on the CEC 2017 test suite, considering problem dimensions of 10, 30, 50, and 100. The optimization results demonstrate that TVETBO exhibits high capabilities in exploring, exploiting, and maintaining a balance between exploration and exploitation during the search process. As a result, it is found that TVETBO offers efficient solutions for benchmark functions.

When compared to twelve popular metaheuristic algorithms, TVETBO performs better than the majority of them when it comes to solving benchmark functions. In general, the suggested TVETBO technique outperforms the competing algorithms in terms of results and performance, as demonstrated by the statistical analysis and simulation results.

To further assess the effectiveness of TVETBO in real-world applications, the algorithm is implemented on twenty-two constrained optimization problems from the CEC 2011 test suite. The simulation results reveal that TVETBO exhibits effective and superior performance in solving constrained optimization problems in real-world applications when compared to competitor algorithms.

Inspiration for TVETBO

Individuals who seek technical and vocational education and training to acquire particular skills referred to as "applicants" are used by this algorithm. Every candidate in TVETBO is a potential fix for the optimization issue, and their characteristics are encoded as values for decision variables.

The mathematical representation of TVETBO involves treating each applicant as a member of a population, and their positions in the search space are modeled using vectors. In this context, a vector is used to represent each applicant's decision variables, where each element of the vector corresponds to a specific decision variable. The entire population of TVETBO members can then be represented as a matrix based on Equation (30). Equation (31) is used to randomly initialize each applicant's position in the search space.

$$A = \begin{bmatrix} A_1 \\ \vdots \\ A_i \\ \vdots \\ A_N \end{bmatrix}_{N\times m} \begin{bmatrix} a_{1,1} & \dots & a_{1,j} & \dots & a_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ a_{i,1} & \dots & a_{i,j} & \dots & a_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ a_{N,1} & \dots & a_{N,j} & \dots & a_{N,m} \end{bmatrix}_{N\times m} \quad (30$$

$$a_{i,d} = lb_d + r\,.\,(ub_d - lb_d), \quad (31)$$

Here, A represents the TVETBO population matrix, A_i denotes the i th applicant (candidate solution), $a_{i,d}$ is its d th dimension in the search space (decision variable), N denotes the number of population members, m denotes the number of decision variables, r is a random number within the interval [0, 1], and lb_d and ub_d indicate the decision variable's lower and upper bounds, respectively.

It is possible to assess the objective function of the problem that corresponds to each TVETBO member's suggested values for the decision variables. To express the set of evaluated values for the objective function, use the A vector that corresponds to Equation (3).

$$\mathrm{F} = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(A_1) \\ \vdots \\ F(A_i) \\ \vdots \\ F(A_N) \end{bmatrix}_{\mathrm{N}\times 1}, \qquad (32)$$

In this case, the evaluated objective function based on the i th applicant is denoted by F_i, and F is the vector of the evaluated objective function.

The appropriate criterion to compare the caliber of population members in offering potential remedies is the assessed values for the problem's objective function. In this sense, the member with the highest assessed value for the objective function is the best, and the member with the lowest evaluated value for the objective function is the worst. Therefore, in each iteration, the best member should be updated based on a comparison of the new values obtained for the goal function.

TVETBO's Mathematical Modelling

The recommended TVETBO approach employs an iteration-based algorithm. Due to its design, the positions of population members are updated in three phases, which replicates the steps involved in getting vocational and technical training. The text that follows describes the process for shifting the applicants' positions inside the search area.

Phase 1: Education in Theory (Exploration)

Instructors in technical and vocational schools make an effort to familiarize students with abstract ideas related to skills. The places of the population members are changed throughout the first TVETBO phase by the applicants' learning of the skills theory from the teacher. TVETBO members' positions in the problem-solving arena have improved as a result of this procedure since the applicant's skills and knowledge can be further enhanced by the instructor's training.

This approach causes significant shifts in the population members' placements, which boosts the algorithm's ability to explore the problem-solving space globally. As a result, improves the algorithm's capacity to investigate the problem-solving space globally. The TVETBO design, which is predicated on the simulation of matching the applicants' knowledge levels to the instructor's knowledge level, therefore modifies the placements of the algorithm population members in the problem-solving space. The trainer is regarded as the best member of the TVETBO design. Equation (33) is used for each candidate to determine a new position based on the interactions between the applicants and the trainer during the training.

Then, in accordance with Equation (34), this new position takes the place of the relevant member's prior position if the value of the objective function is enhanced.

$$a_{i,d}^{P_1} = a_{i,d} + r\,.\left(I_d - S.a_{i,d}\right), \quad i = 1,2,\ldots,N, \quad d = 1,2,\ldots,m, \qquad (33)$$

$$A_i = \begin{cases} A_i^{P_1}, & F_i^{P_1} < F_i \\ A_i, & else \end{cases}, \qquad (34)$$

Based on a simulation of matching the applicant's knowledge level with the instructor, $A_i^{P_1}$ is the new suggested position of the i th applicant (i.e., a TVETBO member as a possible solution for the given problem), where $F_i^{P_1}$ is its objective function value that is determined by inserting the values of the decision variables proposed by $A_i^{P_1}$ in the problem's objective function. $a_{i,d}^{P_1}$ is its d th dimension, denoting the new value for the d th decision variable proposed by the i th TVETBO member, S is a random number from the set {1, 2} that expresses the speed at which the applicants act when learning from the instructor; N is the number of applicants; and m is the number of decision variables, r is a random number with a normal distribution in the range of [0, 1] that is used to create a random nature in TVETBO's performance.

Phase 2: Research and Instruction in Practice

Following the theoretical instruction, the trainer attempts to impart to the candidates in training workshops scientific knowledge of technical and practical skills. The population is updated in the second TVETBO phase by simulating this procedure. Similar to how an instructor can help a novice candidate become a professional over time, this method is used to strengthen TVETBO members' positions in the problem-solving arena so they can find better solutions. Modeling this process causes significant shifts in population members' placements, which boosts the algorithm's capacity for discovery and helps it control the global search inside the problem-solving area. Equations (35) and (36) are used to calculate a new position for each candidate based on how they mimic the teacher when learning technical and vocational skills. Then, in accordance with Equation (37), this new location takes the place of the associated member's prior position if the value of the goal function is increased.

$$K(t) = r.\frac{t}{T} \quad (35)$$

$$a_{i,d}^{P_2} = I_d + K(t)\left(a_{i,d} - I_d\right), \; i = 1,2,\dots,N, \; d = 1,2,\dots,m, and \; t = 1,2,\dots,T \quad (36)$$

$$A_i = \begin{cases} A_i^{P_2} & , F_i^{P_2} < F_i \; ; \\ A_i & , else \quad , \end{cases} \quad (37)$$

Here, $A_i^{P_2}$ represents the new suggested position of the i th applicant (a TVETBO member as a candidate solution for the given problem) based on mimicking the applicant's imitation when learning practical skills from the instructor; $a_{i,d}^{P_2}$ is its d th dimension, denoting a new value for the d th decision variable proposed by the *i* th TVETBO member; and $F_i^{P_2}$is its objective function value, which is determined by inserting the values of the decision variables proposed by $A_i^{P_2}$ in the problem's objective function, A larger K(t) (maximum value of 1) denotes an improvement in the applicant's practical skills. *t* is the algorithm's iteration counter, and T is the maximum number of algorithm iterations. K (t) is the practical education imitation coefficient that is acquired during the educational period.

Phase Three: Strengthening Individual Capabilities

After finishing theoretical and scientific studies at technical and vocational education and training institutes, applicants try to improve their skills to carry out their professions more successfully. TVETBO's third step involves updating the algorithm population through the simulation of individual skill progress. TVETBO members' minor beneficial changes in the problem-solving environment are achieved through this approach since applicants' individual efforts and workouts can only slightly improve their abilities.

When this process is modeled, the population members' placements somewhat alter, which boosts the algorithm's exploitation capability for local searches in the problem-solving space. Thus, a new position is determined for each application using Equation (38), which is based on the modelling of the applicants' effort to improve their performance. Equation (39), then, states that this new location takes the place of the relevant member's prior position if the value of the objective function is improved.

$$a_{i,d}^{P_3} = a_{i,d} + (1 - 2r).\frac{ub_d - lb_d}{t}, \; i = 1,2,\dots,N, \; d = 1,2,\dots,m, and \; t = 1,2,\dots,T, \quad (38)$$

$$A_i = \begin{cases} A_i^{P_3} & , F_i^{P_3} < F_i \; ; \\ A_i & , else \quad , \end{cases} \quad (39)$$

Here, $A_i^{P_3}$ represents the new suggested position of the i th applicant (i.e., TVETBO member as the candidate solution for a given problem) based on simulating the individual skill improvements of applicants over time; $a_{i,d}^{P_3}$ is its (d) th dimension, which denotes a new value for the d th decision variable proposed by the i th TVETBO member; $F_i^{P_3}$ is its objective function value, which is determined by inserting the values of the decision variables proposed by $A_i^{P_3}$ into the problem's objective function; (t) is the algorithm's iteration counter; and T is the maximum number of algorithm iterations. The decision variable d th has two bounds: lb_d and ub_d, which represent its upper and lower limits, respectively. The optimization problem's constraints include the values of the upper and lower bounds; the description of the problem's mathematical model contains this information.

TVETBO Repetition, Pseudo Code, and Flowchart

At the completion of the first TVETBO iteration, the search space placements of every TVETBO member have been modified, taking into account Phases 1 through 3. Then, using Equations (33)–(39), the TVETBO members' positions in the search space are updated until the algorithm's last iteration, at which time the algorithm starts the next iteration using the changed values. At the end of every cycle, the best candidate solution discovered is updated and preserved. The best candidate solution for the problem found during the algorithm's iterations is given once TVETBO has finished being implemented.

Pseudo code of TVETBO #4 [22]

Begin TVETBO.

1. Enter the variables, objective function, and restrictions for the task.
2. Define the number of TVETBOs (N) and the number of iterations (T).
3. Using Equation (31) generate the first population matrix at random.

$$a_{i,d} \leftarrow lb_d + \mathrm{r}\cdot (ub_d - lb_d)$$

4. Analyze the function of the objective.
5. For t = 1 to T
6. For i = 1 to N
7. Phase 1: Education in Theory (Exploration)
8. Update the instructor with the best member of the population.
9. Determine the *i* th TVETBO member's new position by applying Eq (33).

$$a_{i,d}^{P_1} = a_{i,d} + r\,.\left(I_d - S.a_{i,d}\right)$$

10. Use Eq. (34) to get the *i* th TVETBO member.

$$A_i = \begin{cases} A_i^{P_1} & , F_i^{P_1} < F_i \; ; \\ A_i & , else \quad , \end{cases}$$

11. Phase 2: Research and Instruction in Practice
12. Determine the instructor's skill's imitation coefficient by utilizing Eq (35).

$$K(t) = r.\frac{t}{T}$$

13. Using Eq. (36), determine the (i) th TVETBO member's new position.

$$a_{i,d}^{P_2} = I_d + K(t)(a_{i,d} - I_d)$$

14. Utilizing Eq. (37), update the (*i*) th TVETBO member.

$$A_i = \begin{cases} A_i^{P_2} & , F_i^{P_2} < F_i \ ; \\ A_i & , else \quad , \end{cases}$$

15. Phase Three: Strengthening Individual
16. Using Eq. (38), get the *i* th TVETBO member's new position.

$$a_{i,d}^{P_3} = a_{i,d} + (1-2r).\frac{ub_d - lb_d}{t}$$

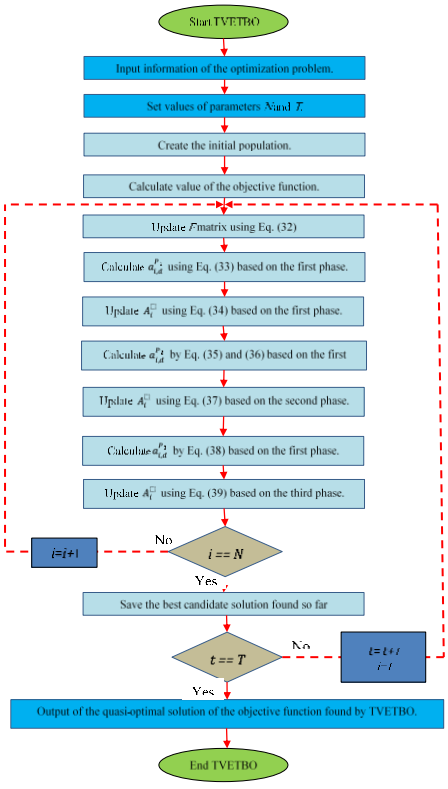
17. Use Eq. (39) to get the *i* th TVETBO member.

$$A_i = \begin{cases} A_i^{P_3} & , F_i^{P_3} < F_i \ ; \\ A_i & , else \quad , \end{cases}$$

18. End
19. Keep the best possible candidate solu for now.
20. End
21. Use the TVETBO, get best optimal solution

End TVETBO.

Figure (4): The TVETBO flow chart



The advantages of TVETBO are:

1- Enhanced efficiency and resource utilization.
2- Improved relevance to industry needs.
3- Personalized learning experiences for students.
4- Stronger industry connections and opportunities.
5- Data-driven decision-making for program improvements.
6- Cultivation of a culture of continuous improvement.
7- Promotion of accessibility and inclusivity in education.

The limitations of TVETBO are:

1-Dependency on technology infrastructure.
2-Privacy and security concerns with handling sensitive data.
3-Potential for bias in algorithms.
4-High costs of implementation and maintenance.
5-Resistance to change from stakeholders.
6-Limited customization options.
7-Reliance on accurate data for effectiveness.
8-Risk of overemphasizing employment outcomes.

Applications of TVETBO:

1- Optimization problem solutions with dimensions of 10, 30, 50, and 100 are assessed use the CEC 2017 test suite. [22]
2- The CEC 2011 test suite contains twenty-two limited optimization problems on which TVETBO is implemented. [22]

6. SEWING TRAINING-BASED OPTIMIZATION (STBO) [23]

A relatively new optimization technique called Sewing Training-based Optimization (STBO) simulates how people learn to sew. Researchers presented the algorithm in 2020, and it has demonstrated good performance in resolving a range of optimization issues.

The STBO algorithm draws inspiration from the way people learn to sew. A person learns to sew by making mistakes, and with time, they get better at it. Similar to this, the STBO algorithm generates a population of solutions, and each one is enhanced by drawing on the knowledge of its predecessors.

The steps of the STBO algorithm are as follows:

1. Initialization: A random population of solutions is produced.
2. Evaluation: The fitness function of each solution is used to determine its value.
3. Selection: Using their fitness value as a guide, the best options are chosen.
4. Learning: New solutions are learned by applying the chosen solutions. To accomplish this, two solutions are chosen at random and combined to

produce a new solution. After evaluation, the new answer is included into the population.

5. Termination: An algorithm comes to an end when a certain number of iterations is reached or a workable solution is obtained.

When it comes to function optimization, feature selection, and data clustering, the STBO algorithm has demonstrated encouraging outcomes. Nonetheless, the problem being addressed and the settings applied determine how well it performs, just like any other optimization process.

In conclusion, a novel optimization method that draws inspiration from the way people learn to sew is called Sewing Training-based Optimization (STBO). It has demonstrated encouraging outcomes in resolving a variety of optimization issues and merits more investigation.

STBO's Mathematical Model

The STBO population can be mathematically represented as a matrix, and the individual STBO members as vectors. The STBO population is represented by a matrix in Equation (40)

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \dots & x_{1,j} & \dots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{i,1} & \dots & x_{i,j} & \dots & x_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ x_{N,1} & \dots & x_{N,j} & \dots & x_{N,m} \end{bmatrix}_{N\times m}, \quad (40)$$

where X_i is the i th STBO member, X is the STBO population matrix, N is the number of STBO population members, and m is the number of problem variables. Equation (41) is used to initialize all population members at random at the start of the STBO implementation.

$$x_{ij} = lb_j + r\,.\,(ub_j - lb_j),$$
$$i = 1,2,\dots,N, \qquad j = 1,2,\dots,m, \quad (41)$$

The j th problem variable's lower and upper bounds are denoted by lb_j and ub_j respectively, and x_{ij} is the variable's value as decided by the i th STBO's member X_i, r is a random number within the interval [0, 1].

Equation (42) use a vector to model the values generated for the target function, taking into account the relative positions of the candidate solutions in the issue variables.

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1}, \quad (42)$$

where F_i is the objective function value for the i th candidate solution and F is the objective function vector.

The solution with the best value for the objective function is identified as the best candidate solution, or the best individual in the population X_{best}.

Updating candidate answers in STBO involves three steps: practice, imitation of the instructor's techniques, and instruction.

First Phase: Exploration and Training

The collection of all potential members serving as potential trainers for every STBO member X_k , $i = 1,2,\dots,N$ is defined using the following identity

$$CSI_i = \{X_k | F_k < F_i\,,\ k \in \{1,2,\dots,N\}\} \cup \{X_{best}\}, \quad (43)$$

In this case, CSI_i represents the collection of all potential candidate training instructors for the i STBO member.

X_{best} is the only potential candidate training instructor in this scenario $X_i = X_{best}$, or alternatively $CSI_i = X_{best}$. The training instructor of the i th member of STBO is then chosen at random from the set $CSI_i$for each i ∈ {1, 2… N}; this member is designated as SI_i. i th STBO member is taught sewing skills by this chosen instructor SI_i.

Equation (44) is used to construct a new location for each population member before updating them depending on this phase of the STBO.

$$x_{i,j}^{P_1} = x_{i,j} + r_{i,j}\,.\,(SI_{i,j} - I_{i,j}\,.\,x_{i,j}), \quad (44)$$

Where $I_{i,j}$ are randomly selected numbers from the set {1, 2}, $r_{i,j}$ are random numbers from the interval [0, 1], $x_{i,j}^{P_1}$ is its d th dimension, and $F_i^{P_1}$ is its objective function value.

Eq. is used to model this update condition (45)

$$X_i = \begin{cases} X_i^{P_1} & , F_i^{P_1} < F_i\ ; \\ X_i & , otherwise\ , \end{cases} \quad (45)$$

where $X_i^{P_1}$ represents the i th STBO member's new according to the STBO's first phase.

Phase 2: involves emulating the instructor's techniques (exploration).

Every decision variable in this STBO phase is thought to represent a sewing skill. Every STBO member mimics the selected instructor's (ms) skills, $1 \le ms \le m$.

During this process, the population of the algorithm is transferred across the search space, illustrating the exploratory capabilities of the STBO. The collection of variables (i.e., the set of training instructor's skills) that each STBO member imitates is specified in equation (46).

$$SD_i = \{d_1, d_2, \dots, d_{m_s}\}, \quad (46)$$

With $m_s = \left[1 + \frac{t}{2T}m\right]$ denoting the number of skills chosen to mimic, t the iteration counter, and

T the total number of iterations, SDi is an(ms) – a combination of the set {1, 2, …, m }, which represents the set of the indexes of decision variables (i.e., skills) identified to imitate by the i th member from the instructor.

The simulation of reproducing these instructor skills is used to determine each STBO member's new position using the following identification.

$$x_{i,j}^{P_2} = \begin{cases} SI_{i,j} & , \ j \in SD_i \ ; \\ x_{i,j} & , \ otherwise \ , \end{cases} \qquad (47)$$

Where $x_{i,j}^{P_2}$ is the d th dimension of $X_i^{P_2}$ and $X_i^{P_2}$is the newly produced location for the i th STBO member based on the second phase of STBO. If this new position increases the value of the objective function, the relevant member's old position is replaced.

$$X_i = \begin{cases} X_i^{P_2} & , \ F_i^{P_2} < F_i \ ; \\ X_i & , \ otherwise \ , \end{cases} \qquad (48)$$

Where $F_i^{P_2}$ the value of is $X_i^{P_2}$'s objective function.

Phase 3: Application (utilization)

This STBO phase illustrates the suggested algorithm's suitability for local search. Before mathematically modelling this STBO phase, equation (49) is used to establish a new location around each STBO member (with an adjustment to maintain all newly calculated population members in the chosen search region).

$$x_{i,j}^{P_3} = \begin{cases} lb_j \ , \ x_{i,j}^* < lb_j \ ; \\ x_{i,j}^* , \ x_{i,j}^* \in [lb_j \ , ub_j \ ] \ ; \\ ub_j \ , \ x_{i,j}^* > ub_j \ , \end{cases} \qquad (49)$$

Where $x_{i,j}^* = x_{i,j} + (lb_j + r_{i,j} \ . (ub_j - lb_j)/t$ and the random number $r_{i,j}$ comes from the interval [0, 1]. Equation (50) then states that if the objective function's value increases, the STBO member's prior position will be replaced.

$$\mathrm{X_i} = \begin{cases} \mathrm{X_i^{P_3}} & , \ \mathrm{F_i^{P_3}} < \mathrm{F_i} \ ; \\ \mathrm{X_i} & , \ \mathrm{otherwise} \ , \end{cases} \qquad (50)$$

Since the second phase of STBO generates the location for the i th STBO member, $X_i^{P_3}$ represents its d th dimension, and $F_i^{P_3}$ is the value of its objective function.

STBO's Repetition Process and Pseudo-Code

After then, the update process is carried out until the algorithm's last iteration, which is based on Equations (43) through (50). When the STBO is fully applied to the provided problem, the solution that stands out as the best candidate during the algorithm iteration is displayed. Lastly, the STBO implementation phases' pseudo-code is shown in Algorithm 1.

The STBO Pseudo-Code #5. [23]

Begin STBO.

1. Enter the variables, restrictions, and objective
2. Determine the STBO's population size (N) and number of iterations (T).
3. Set the STBO population's initial values using equation (41) and construct vector F with the goal function's values using equation (42).
4. For t = 1 to T
5. For i = 1 to $N_{\square}$
6. First Phase: Exploration and Training
7. Determine the set of candidate training instructors for the i th member by eq. (43). $CSI_i \leftarrow \{X_K | F_K < F_i, k \in \{1,2,\dots,N\}\} \cup \{X_{best}\}$.
8. Select the SI_i training from CSI_i to instruct the i th STBO member in sewing.
9. Determine the i th STBO member's new position by utilizing (44).

$$x_{i,j}^{P1} \leftarrow x_{i,j} + r_{i,j} \ . (SI_{i,j} - I_{i,j} . x_{i,j})$$

Update the i th STBO member's position with (45).

$$X_i \leftarrow \begin{cases} X_i^{P1} \ , & F_i^{P1} < F_i \\ X_i \ , & else \end{cases}$$

10. Phase 2: Exploration and imitation of the instructor's skills
11. With Equation (46), determine SD_i.
12. Determine the i th STBO member's new location by applying Equation (47).

$$\mathrm{x_{i,j}^{P2}} \leftarrow \begin{cases} \mathrm{SI_{i,j}} \ , & \mathrm{j \in SD_i}; \\ \mathrm{x_{i,j}} \ , & \mathrm{else.} \end{cases}$$

13. Apply Eq (48) to update the i th STBO member's position.

$$X_i \leftarrow \begin{cases} X_i^{P2} \ , & F_i^{P2} < F_i \ ; \\ X_i \ , & else \end{cases}$$

14. Phase 3: Practice (exploitation)
15. Calculate the new position for the i th STBO member using (49).

$$X_{i,j}^{P3} \leftarrow x_{i,j} + \frac{lb_j + r_{i,j}(ub_j - lb_j)}{t}.$$

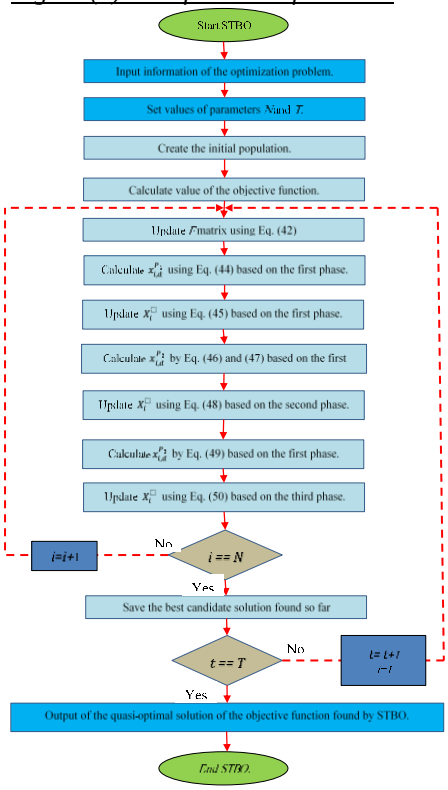
16. Update the i th STBO member's position using Eq (50).

$$\mathrm{X_i} \leftarrow \begin{cases} \mathrm{X_i^{P3}} \ , & \mathrm{F_i > F_i^{P3}} \ ; \\ \mathrm{X_i} \ , & \mathrm{else} \end{cases}$$

17. End.
18. The best candidate solution should be updated.
19. End
20. Output: The best approximate optimal solution found using CBOA.

End STBO.

Figure (5): The flowchart for STBO



The benefits of STBO include:

1- Applications in optimization and solution presenting of the suggested STBO method are evaluated.
2- These six statistical indicators mean, standard deviation (std), best, worst, median, and rank are used to report the outcomes of running metaheuristic algorithms. As a ranking criterion, the mean of rank is used to assess how well optimization algorithms perform in each of the objective functions.
3- The adoption of the STBO sets up a number of projects for the following studies.
4- Binary and multimodal versions are among the most targeted study directions suggested by STBO.
5- Using STBO in various practical and scientific optimization applications is another proposal for future research.

The STBO's limitations are

1- Dependence on expertise, which can lead to suboptimal solutions.
2- Sensitivity to initial conditions, potentially causing convergence issues.
3- Computational complexity, especially for large-scale problems.
4- Limited applicability to certain problem types.
5- Challenges in scaling to larger teams or problem domains.
6- Risk of converging to local optima instead of global optima.
7- Continuous need for expert involvement, which may not always be feasible.

STBO applications:

1. Design of pressure vessels (PVD). [9]
2. Design of a speed reducer (SRD). [13], [14]
3. Design of welded beams (WBD). [10]
4. Design of tension/compression springs (TCSD). [12]

7. Volleyball Premier League Algorithm (VPL) [24]

The suggested VPL algorithm simulates how teams interact with one another in a league of volleyball. Additionally, it documents the coaching decisions made during a match. The main source of inspiration for the proposed algorithm is mentioned first, followed by the mathematical model.

A new game called Mintonette was introduced by William G. Morgan more than a century ago, in 1895. Due to the volleying aspect of the game, volleyball was adopted as the new name in 1896[110]. A net divides two teams of six players during a match. To score a point, each team must in specific, regulated circumstances ground the ball on the other team's court. The way the game went was as follows: a player from one team would attempt to spark a rally by serving the ball over the net and into the opposing team's court. The rally will go on until one team makes a mistake, at which point another team will score a point. A team's head coach sends the starting lineup to the second. Based on their personnel, each side is able to put together a unique starting lineup. During the game, the coach typically stands close to the court to guide the players. While they get instruction and guidance from the coach and analysts, substitutes remain on the bench, the contacts between the coach and the team members are threefold. The suggested approach incorporates unique operators to capture these interactions.

Mathematical Model of VPL

There are three types of team members: players, substitutes, and the coach, as was previously indicated. The structure of the solution representation in the suggested algorithm is unique compared to existing metaheuristic algorithms. The active and passive portions are the two segments that make up the solution representation. Six active players make up each team's core, which is represented by the active portion. This section is used to calculate each solution's fitness function. A few variables are held in the passive section to provide unique inspiration rules, such as substitution strategy. In actuality, the head coach may decide to assign

a substitute to take the spot of the player who is leaving the court. Referees have given their approval for this substitution, which takes place.

Similar to other optimization algorithms, the VPL algorithm begins by initializing the teams, which stand for the collection of preliminary fixes for the issue. Every squad possesses two primary characteristics, as previously stated: their formation and their replacements. By using the population of NT teams, where NT represents the number of teams (population size) in each season $TEAM^O$defines the original set of teams. For now, let X_j^f andX_j^srepresent the formation and substitute qualities of the j th variable, respectively, and use Eqs. (51) and (52) to assign random integers between each variable's lower and upper bounds.

$$X_j^f = lb_j + Rand( ) \times (ub_j - lb_j) \quad (51)$$

$$X_j^s = lb_j + Rand( ) \times (ub_j - lb_j) \quad (52)$$

The variables' lower and upper bounds are denoted by the letters lb_j andub_j, respectively. Rand () provides an evenly distributed random number between 0 and 1. Matrix F and S serve as a good example of the basic characteristics of initial solutions. One can determine the number of dimensions and the number of teams (NT) by counting the rows and columns. The NT value can be selected with care, even while the number of variables and dimensions remain the same. Eqs. (53) And (54) define the matrices that capture the formation and substitutes properties of the teams, respectively.

$$F = \begin{bmatrix} X_{1,1}^f & X_{1,2}^f & \dots & X_{1,j}^f \\ X_{2,1}^f & X_{2,2}^f & \dots & X_{2,j}^f \\ \vdots & \vdots & \ddots & \vdots \\ X_{i,1}^f & X_{i,2}^f & \dots & X_{i,j}^f \end{bmatrix} \quad (53)$$

$$S = \begin{bmatrix} X_{1,1}^S & X_{1,2}^S & \dots & X_{1,j}^S \\ X_{2,1}^S & X_{2,2}^S & \dots & X_{2,j}^S \\ \vdots & \vdots & \ddots & \vdots \\ X_{i,1}^S & X_{i,2}^S & \dots & X_{i,j}^S \end{bmatrix} \quad (54)$$

A game schedule is necessary for any conventional sports tournament. Sports scheduling has gained attention in the computer science and operations research areas in recent years [111].A meta-heuristic algorithm must produce game schedules in order to have an optimized solution. We make the league schedule artificially using the single round robin (SRR) approach. In this step, each team plays each other exactly once in a single round. Every time the same two teams play each other again in two straight rounds, there is a repetition of action. One team will not play in each round if the number of teams is not equal. To ensure that the number of teams is even, we can add a dummy team.

We give mathematical formulae indicating a team's possibility of winning and power to win the match in order to determine which team will win the competition. We assume that linear equations produced the relationship between each team's powers. In this case, we're assuming that team tournaments are idealized and that the outcome of each match is unaffected by unforeseen circumstances. We present the team power index as follows in order to ascertain the creation of the team i designated as X_i^f as well as the strength of each team in a given week:

$$\varphi(i) = \frac{f(X_i^f)}{Z} \quad (55)$$

$$Z = \sum_{i=1}^{n} f(X_i^f) \quad (56)$$

Where Z represents the total sum of fitness values over the course of a week, $f(X_i^f)$ represents the fitness value of team i, and $\varphi(i)$ represents the power index for the team. According to Equation (55), the team power is inversely related to each team's fitness value. The value of $\varphi(i)$ rises with team strength. The fitness value can be used to determine a team's probability of winning a match because each team is weighed differently. Assume that there are two teams and k participating in a contest. Their respective formations are X_i^f and X_k^f. For teams, the power index is therefore calculated as follows:

$$\varphi(i) = \frac{f(X_j^f)}{Z} \quad (57)$$

$$\varphi(k) = \frac{f(X_k^f)}{Z} \quad (58)$$

Let the probability that team j wins the match be represented by p (j, k). Next up, we have

$$p(j,k) = \frac{\varphi(j)}{\varphi(j)+\varphi(k)} \quad (59)$$

The following probability principle applies to each match between teams k and j:

$$p(j,k) + p(k,j) = 1 \quad (60)$$

The following formula is also true:

$$p(k,j) = 1 - \frac{\varphi(j)}{\varphi(j)+\varphi(k)} \quad (61)$$

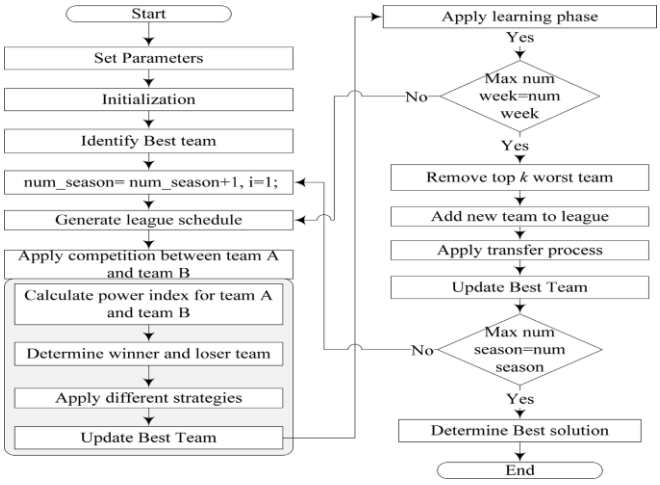
Since p (j, k) indicates the likelihood of winning a match, the winner of any match may be determined using a uniformly distributed random number, r ∈ [0, 1]. If r < p (j, k), then team J wins; otherwise, team K wins. It is evident that if $f(X_j^f)$ and $f(X_k^f)$ are close to one other, then p (j, k) and p (k, j) tend to be 0.5. After the fight is settled, strategies for the winning and losing sides are employed to create a new structure. While the losing team can investigate information exchange, relocation, and substitution, the victorious team can utilize the leading role method.

Team I and Team J have a pseudo-code of competition.

Pseudo-code of VPL #6 [25]

Function Competition (i, j)
  Calculate $\varphi(i)$ and Z using (55) and (56)
  Calculate $p(i,j)$ usin
  Generate $r \in [0,1]$, (59)
  If $p(i,j) \leq r$
    team i is the winner and team j is the
  Else
    team j is the winner and team i is the
  End if
  apply winning strategy for the winner,
  apply losing strategies for the loser,
End

Figure (6): The VPL flow chart.



The advantages of VPL are:

1- Fairness in scheduling and ranking.
2- Efficient scheduling with minimized conflicts.
3- Flexibility to accommodate changes.
4- Enhanced fan engagement with competitive matchups.
5- Encourages player and team development.
6- Data-driven decision-making for optimization.
7- Reduced bias in decisions.
8- Cost-effectiveness in resource allocation.
9- Scalability to fit various league sizes.
10- Continuous improvement through feedback and refinement.

The limitations of VPL are:

1- Complexity in implementation.
2- Dependency on resources like data accuracy.
3- Potential oversight of human factors.
4- Risk of algorithmic bias.
5- Resistance to change from traditional methods.
6- Limited adaptability to rapid changes.
7- Potential for over-optimization.
8- Technical challenges in maintenance.
9- Scalability concerns.
10- Ethical considerations regarding fairness and transparency.

Applications of VPL:

1- Solved the three typical optimization problems in the field of designing engineering.[25]
2- use an alternative multilevel thresholding image segmentation method.[26]
3- Solving the aforementioned problem for green scheduling identical parallel machines with splitting jobs.[27]
4- Solve Global Optimization with the Engineering Problems.[28]

7. The Election-Based Optimization Algorithm (EBOA) [29]

The Election-Based Optimization Algorithm (EBOA), a novel optimization algorithm, was created to replicate the leader-selection voting procedure. The main sources of inspiration for EBOA were the election process, the selection of the leader, and the impact of citizens' awareness on that decision. The search space, which is led by the chosen leader, directs the EBOA population. The two stages of the EBOA process exploration and exploitation are represented numerically. The efficiency of EBOA has been examined in 33 objective function examples of different kinds (uni-modal, high dimensional multi-modal, fixed-dimensional multi-modal, and CEC 2019). The implementation results of the EBOA on the goal functions show that it can effectively explore and exploit both local and global search spaces while striking the right balance between them. These qualities have contributed to the suggested EBOA approach's effectiveness in optimizing and delivering pertinent solutions. Based on our investigation, we have found that EBOA outperforms the 10 other algorithms that it was matched against by offering an acceptable balance between exploration and exploitation.

Initialization for EBOA

The members of the population-based metaheuristic algorithm EBOA are members of the community. Every person in the population symbolizes a suggested fix for the issue in the EBOA. From a mathematical perspective, the population of EBOA is represented by a matrix using Eq. (62) that is referred to as the population matrix.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \dots & x_{1,j} & \dots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{i,1} & \dots & x_{i,j} & \dots & x_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ x_{N,1} & \dots & x_{N,j} & \dots & x_{N,m} \end{bmatrix}_{N\times m}, \quad (62)$$

where N is the EBOA population size, m denotes the number of decision variables, X_i denotes the i th EBOA member (i.e., the suggested solution), and $x_{i,j}$ denotes the value of the j th problem variable provided by the i th EBOA member.

According to Equation (63), the starting positions of each individual in the search space are chosen at random.

$$X_{i,j} = lb_j + r\,.\,(ub_j - lb_j), \quad i = 1,2,\dots,N \;\; j = 1,2,\dots,m \quad (63)$$

The variables lb_j and ub_j denote the lower and upper bounds of the jth variable, respectively, while r represents a random number inside the interval [0; 1].

Based on what each EBOA member offers as a value for the problem variables, an objective function value can be assessed.

These evaluated values are given by equation (64) for the objective function of the issue using a vector.

$$OF = \begin{bmatrix} OF_1 \\ \vdots \\ OF_i \\ \vdots \\ OF_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} OF(X_1) \\ \vdots \\ OF(X_i) \\ \vdots \\ OF(X_N) \end{bmatrix}_{N\times 1}, \quad (64)$$

where OF_i denotes the achieved objective function value for the ith EBOA member and OF refers to the vector of obtained objective function values of the EBOA population. The objective function's values serve as a criterion for evaluating the suggested solutions' quality, with the best value of the function designating the best member and the worst value designating the worst member.

Mathematical Modeling of EBOA

The method that updates population members and refines suggested solutions with each iteration is the primary distinction amongst metaheuristic algorithms. The two phases of exploration and exploitation that make up the process of upgrading the algorithm population in EBOA are covered here.

Phase 1: Election procedures and voting (examination).

Members of EBOA take part in the election and cast their votes for a candidate based on their level of knowledge. People's awareness is determined by the standard and goodness of the objective function's value. As a result, Eq. (65) is used to imitate community members' awareness. Better values of the objective function indicate greater awareness in this awareness simulation procedure.

$$A_i = \begin{cases} \frac{OF_i - OF_{worst}}{OF_{best} - OF_{worst}}, & OF_{best} \neq OF_{worst}; \\ 1, & else, \end{cases} \quad (65)$$

where OF_{best} and OF_{worst} are the best and worst values of the objective function, respectively, and Ai is the awareness of the i th EBOA member. It should be emphasized that OF_{best} in minimization problems is correlated with the objective function's minimal value and OF_{worst} maximum value; in maximisation problems, on the other hand, OF_{best} is related to the objective function's maximum value and OF_{worst} to its minimum value.

Ten percent of the most conscious members of the society are thought to be potential contenders for office. It is expected in the EBOA that the minimum number of candidates (NC) is two, or $NC \geq 2$. This implies that a minimum of two candidates will file to run in the election.

The way the EBOA voting method is implemented compares each person's awareness level to a random number; if an individual's awareness level is higher than that random number, they are eligible to vote for the top contender, C_1. If not, that individual votes for one of the other contenders at random. Equation (66) provides a mathematical model of this voting procedure.

$$V_i = \begin{cases} C_1, & A_i > r; \\ C_k, & else, \end{cases} \quad (66)$$

where V_i denotes the community member's vote for the i th time, C_1 denotes the best candidate, and C_k denotes the k th contender, k being a randomly chosen number from the range $\{2,3,\dots, N_C\}$.

After the votes are counted, the candidate with the most votes at the conclusion of the contest is declared the elected (leader). Each person's situation within society, including those who did not vote for him, are impacted by this chosen leader. Under the direction and influence of the elected leader, each member's status within the EBOA is updated. This leader improves the EBOA's capacity for global search exploration by guiding the algorithm population to various regions in the search space. The process of updating the EBOA population, which first generates a new position for each member, is supervised by the leader. If updating to the newly created position

increases the objective function's value, then it is acceptable. In the event that not, the appropriate member holds their prior position. The EBOA update process is modelled by means of Equations (67) and (68).

$$x_{i,j}^{new,P_1} = \begin{cases} x_{i,j} + r.(L_j - I.x_{i,j}) & , OF_L < OF_i ; \\ x_{i,j} + r.\left(x_{i,j} - L_j\right) & , \quad else \quad , \end{cases} \quad (67)$$

$$X_i = \begin{cases} X_{i,j}^{new,P_1} & , OF_i^{new,P_1} < OF_i ; \\ X_i & else, \end{cases} \quad (68)$$

where I is an integer chosen at random from the values of 1 or 2, L is the elected leader, L_j is its j th dimension, and OF_L is its objective function value. $x_{i,j}^{new,P_1}$ denotes a new generated position for the i th EBOA member.

Phase 2: Public awareness-building initiative (exploitation).

The level of awareness among society's members greatly influences their ability to make wise choices during the election and voting process. Apart from the impact of the leader on individuals' consciousness, each person's ideas and deeds can heighten their awareness. From a mathematical perspective, a local search next to any suggested answer might reveal a superior solution. As a result, community members' efforts to raise awareness boost the EBOA's capacity to leverage local searches and identify better solutions to issues. A random position is taken into consideration in the vicinity of every member in the search space in order to replicate this local search procedure.

After then, the objective function of the problem is assessed in light of the new circumstance to ascertain whether it is preferable to the member's current condition. If the new position has a higher value for the objective function, the local search is successful, and the connected member's position is updated. Enhancing the objective function's value will make that person more conscious, enabling them to make wiser decisions in the upcoming election (or iteration). Eqs. (69) and (70), which model this updating process to raise people's knowledge in the EBOA, are used.

$$x_{i,j}^{new,P_2} = x_{i,j} + (1 - 2r).R.\left(1 - \frac{t}{T}\right).x_{i,j} \quad (69)$$

$$X_i = \begin{cases} X_{i,j}^{new,P_2} & , OF_i^{new,P_2} < OF_i ; \\ X_i & else, \end{cases} \quad (70)$$

Where $X_{i,j}^{new,P_2}$ denotes a newly generated position for the i th EBOA member, $x_{i,j}^{new,P_2}$ denotes its j th dimension, OF_i^{new,P_2} denotes the objective function's value, R denotes a constant equal to 0.02, t denotes the iteration contour, and T denotes the maximum number of iterations.

The Repetition Process, Pseudo Code, and EBOA Flowchart

All population members' statuses are updated, and then an EBOA iteration is finished. Following the first and second phases, the population update procedure is repeated according to Eqs. (65) to (70) until the last iteration, at which point the updated values of the EBOA are entered into the subsequent iteration. The greatest suggestion discovered during the algorithm's iterations is presented by EBOA as the solution to the issue once the method has been fully implemented.

Pseudo code of EBOA #7 [29]

Begin EBOA.

Enter the variables, objective function, and restrict set the number of EBOA iterations (T) and (N).

Create a random beginning population matrix.

Analyze the function of the objective.

For t = 1 to T

Update the population's finest and worst members.

Phase 1: Election procedures and voting.

Apply Eq. (65) to compute A.

Select applicants according to awareness standards.

Create a vote and election simulation with Eq. (66).

Determine who will be the new leader after tallying

for i = 1 to N

Determine X_i^{new,P_1} using Eq. (67).

Update X_i using Eq. (68).

Phase 2: Public movement to raise awareness.

Calculate X_i^{new,P_2} using Eq. (69).

Update X_i using Eq. (70).

End

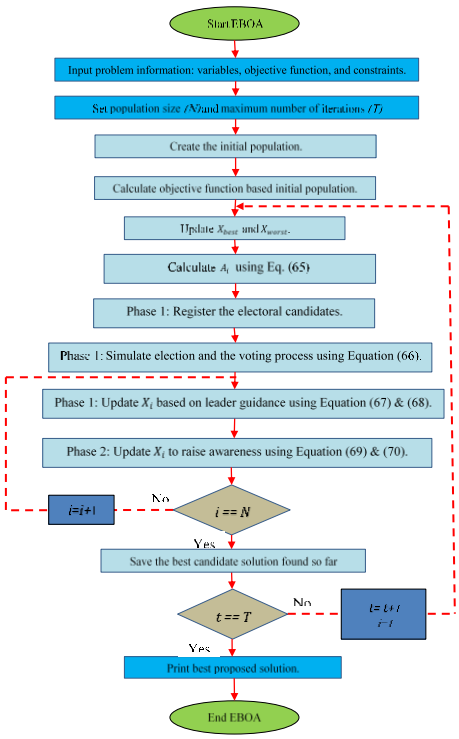
Save best proposed solution so far.

End

Output best quasi-optimal solution obtained with the EBOA.

End EBOA.

Figure (7): The EBOA flow chart [29]



The advantages of EBOA are:

1- Decentralized Coordination: Enable efficient task distribution without a central authority.
2- Fault Tolerance: Quickly detect and recover from node failures.
3- Scalability: Distribute decision-making as the system grows, maintaining performance.
4- Load Balancing: Evenly distribute workload across nodes.
5- Adaptability: Dynamically adjust to changes in network topology.
6- Reduced Communication Overhead: Minimize unnecessary communication.
7- Self-Organization: Enable autonomous adaptation to changing conditions.

The limitations of EBOA are:

1- Complexity in implementation.
2- Increased communication overhead.
3- Susceptibility to attacks.
4- Potential for single points of failure.
5- Consensus delays.
6- Scalability constraints.
7- Vulnerability to churn in dynamic environments.
8- Synchronization challenges.

Applications of EBOA:

1- Machine learning-aided modeling for predicting freshwater production of a membrane desalination system. [30]
2- Pressure vessel design (PVD). [9]
3- Proposed in Wireless Sensor Networks (WSNs) to make sure that the Cluster Head (CH) selection process to identify the best CH. [31]
4- Welded beam design (WBD). [10]
5- Aids in the efficient transmission of medical data within the Internet of Medical Things (IoMT) in a much faster and shortest way. [32]
6- Optimal Feature Selection and Classification of Respiratory Diseases. [33]

7. Interior Search Algorithm (ISA) [34]

"Interior search algorithm" is a broad term that can refer to various algorithms used in computational science and engineering, particularly in optimization and numerical analysis. These algorithms are designed to find solutions within a specified region or domain, often referred to as the "interior" of a feasible space These are just a few examples of interior search algorithms, and there are many more variations and applications depending on the specific problem domain. The choice of algorithm depends on the nature of the problem being solved and the desired properties of the solution, such as efficiency, accuracy, and scalability.

Inspiration Analysis

Amir H. Gandomi (2014) introduced the Interior Search Algorithm (ISA) as a solution to global optimization challenges. The aesthetics of objects and mirrors serve as ISA's primary source of inspiration; more precisely, ISA addresses the mirror and composition groups. Initially, every accessible object is divided into two groups: the composition group and the mirror group. These groups are responsible for determining the best locations for the objects and mirrors, respectively. To obtain a more visually appealing view, the element's composition in the composition group can be changed. Ultimately, the mirrors are positioned as well as they can be between the items to enhance their appearance; the primary goal is to locate the mirror closest to the object.

Mathematical Model of ISA

The aesthetic mechanisms employed in interior design and decoration serve as the foundation for the operation of ISA. The six steps that make up the ISA approach are listed below:

(1) The process begins by randomly generating the locations of the elements, which only apply to the Upper Bound (UB) and Lower Bound (LB). Each random location's fitness is calculated, and the fittest element is chosen as the global best and designated as x_{Gbest}^{j} based on the minimization or maximization problem. Based on the minimization or maximization issue, the fittest element is selected as the global best and identified as x_{Gbest}^{j}. The fitness of each random location is determined.

(2) Moreover, the residual components are arbitrarily divided into two separate categories: composition and mirror. The following rules, in addition to the unique parameter (α), form the basis of this division:

if $r_1 \leq \propto$ so
component ∈ composition
else
aspect Δ mirror
end if

For every element in this case, r_1 is a random value in the interval (0, 1). Selecting α at the right value is crucial to prevent taking a biased stance.

(3) Subsequently, each element inside the composition group should have its composition altered at random using:

$$x_i^j = LB^j + (UB^j - LB^j) \times r_2 \qquad (71)$$

Here, the iteration and element position are denoted by j and i, respectively. r_2 is the value created at random in the interval between 0 and 1, and $LB^j$and UB^j are the lower and upper limit values of the j-th iteration.

(4) A mirror object is first arbitrarily positioned between each element and the global best element of the problem to find the optimal location for the mirrors. The position of the mirror for the i-th item and j-th iteration is given by:

$$x_{mir,i}^j = r_3 \times x_i^{j-1} + (1 - r_3) \times x_{Gbest}^j \quad (72)$$

Where r_3 is a randomly generated variable with a value between 0 and 1, and mir is the mirror object. Furthermore, the following formula can be used to calculate the image location:

$$x_i^j = 2x_{mir,i}^j - x_i^{j-1} \qquad (73)$$

(5) The following describes how to use random walk to change the value of global best:

$$x_{Gbest}^j = x_{Gbest}^{j-1} + r_n \times \lambda \qquad (74)$$

Where a vector of properly distributed random numbers is denoted by r_n. Both x and r_n should have the same dimension.

Here, λ stands for scaling factor, and the problem's search space size has a big impact on how much it is.λ's actual value can be calculated as follows:

$$\lambda = 0.01 \times (UB - LB) \qquad (75)$$

(6) At last, the composition and mirror elements' fitness values are computed. It is necessary to update each location in light of the enhancement. One way to formulate the minimization issue is a

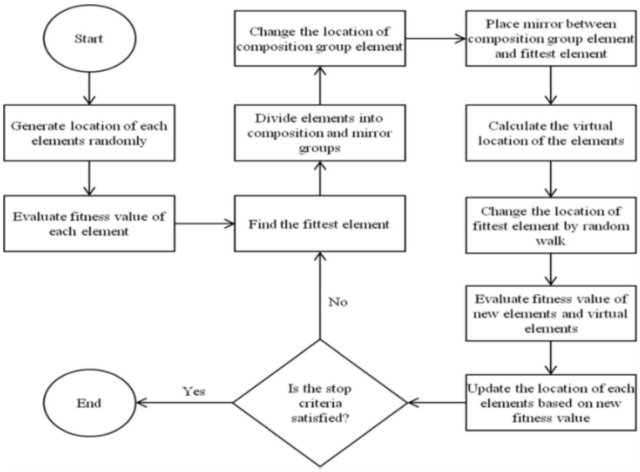
$$x_i^j = \begin{cases} x_i^j & f(x_i^j) < x_i^{j-1} \\ x_i^{j-1} & else \end{cases} \qquad (76)$$

Algorithm: provides the ISA pseudo-code.

An ISA pseudo-code #8.[34]

Input: the fitness function, number of elements, number of iterations, upper and lower bounds.
Output: the ideal resolution
Set the elements' initial values.
while if the stop requirement is not satisfied, do
Find x_{Gbest}^j
for each element do
if x_{Gbest}^j
else if $r_1 \leq ar_1 \leq \propto$
else
end if
Verify the border restrictions.
end for
for every component, do
Compute $f(x_i^j)$
end for
end while

Figure (8): The ISA flow chart [35]



The advantages of ISA are:

1- Efficiency: They converge to solutions quickly, making them suitable for large-scale optimization.
2- Accuracy: They produce highly accurate solutions, often close to the global optimum.
3- Feasibility: Solutions generated satisfy problem constraints, ensuring practicality.
4- Constrained Problems: Well-suited for handling constrained optimization tasks.
5- Robustness: Effective across various problem characteristics without extensive tuning.
6- Scalability: Able to handle large-scale optimization efficiently.
7- Parallelism: Some methods can be parallelized, leveraging modern computing architectures.
8- Versatility: Applicable to diverse problem domains, enhancing their utility in many fields

The limitations of ISA are:

1- Sensitivity to initial conditions.
2- Difficulty with non-convex problems.
3- Complexity and overhead.
4- Memory requirements.
5- Limited applicability to discrete problems.
6- Difficulty with infeasible problems.
7- Algorithmic complexity.
8- Computational cost.
9- Numerical stability issues.

Applications of ISA:

1- Large-Scale Reservoirs System Operation Optimization [35]
2- Validated using both unimodal and multimodal features on a wide selection of 13 benchmark functions. [34]
3- Run this problem using 10 elements and 40 iterations. [36]
4- Optimized Parameters of SOFC for steady state and transient simulations. [37]
5- Various industries have used this algorithm to solve their problems, including power costs and pollutant emissions (Trivedi et al. 2016). [38]
6- Multiobjective optimal power flow. [39]

7. The Social Engineering Optimizer (SEO) [40]

Each solution in this algorithm represents a person, and all the variables of each solution in the search space correlate to the attributes of each individual, such as aptitude for sports, commerce, and mathematics. The algorithm starts with the two random solutions initialized; the more favorable option is called the attacker, while the less favorable solution is called the defender. For every defender trait, a few haphazard trials are created to mimic the attacker's training and retraining from the defender. The attacker aims to assess the defense based on his or her characteristics. The analogue of training and retraining in search space is to copy an attacker's trait to a defender's while simultaneously estimating the attacker's rate of retraining from the defender. The next stage involves detecting a SE attack from the attacker to the defense, which is the same as the defense moving strategically to a feasible location. The best position is selected after evaluating the defender's new position's fitness and comparing it to their previous and current positions when defending against a SE attack. If the defender's skill surpasses the attacker's, both the attacker and the defender trade positions. Ultimately, it is destroyed to attack the defense, and a new random solution takes its place in search space. To facilitate SEO's early phases.

Mathematical Model of SEO:

1. Initialize the attacker and the defender

The optimizer's goal is to identify the optimal option from all feasible options. In this way, a range of variable values that require optimization are produced. For example, in GA, the array is referred to as a "chromosome", whereas in SEO, the term "person" is used instead. Consequently, the individual is the solution's opposite. Additionally, each variable in an array in GA is assigned a "gene" However, "trait" in SEO refers to the individual for every variable. This is covered in several business, sports, mathematical, and other subjects. Persons are $1 \times N_{var}$ arrays in N_{var}-dimensional optimization problems. The definition of this array is:

$$person = [X_1, X_2, X_3, \dots, X_{N_{var}}]. \quad (77)$$

Additionally, two people's Objective Function (OF) values are assessed as follows: $Value = f(person) = f(X_1, X_2, X_3, \dots, X_{N_{var}})$. (78)

2. Train and retrain

This stage tries to demonstrate how the defender has trained and retrained the attacker. In this manner, the attacker attempts to evaluate every quality of the defender in order to identify the most effective quality. In this context, $\propto$ the attacker trait proportion is chosen at random and substituted with identical defense trait percentages as follows:

$$N_{\text{Train}} = round\{\propto . nV\ ar\} \quad (79)$$

Where $\propto$ the proportion of chosen characteristics and $nV\ ar$ is the total number of traits in an individual. As a result, N_{Train} is the number of qualities that the defender's random attributes will be tested on.

3.1. Acquiring

This strategy allows the attacker to use the defender as a direct guide to accomplish their goals. This leads to the creation of a novel solution. This equation is suggested in order to establish the new position:

$$def_{new} = def_{old} \times \left(1 - \sin\beta \times U(0,1)\right) + \frac{def_{old}+att}{2} \times \sin\beta \times U(0,1) \quad (80)$$

Eq. (71) Seeks to depict the defender's movement by taking into account both the attacker's and the defender's average distance from each other. This move is produced based on the random distribution $\boldsymbol{U}.e.$ U (0, 1) and the amount of β as the attack detection rate.

3.2. Phishing

This tactic involves the attacker posing as an approached, after which the defender walks to a location that the attacker desires. This suggested method yields two fresh solutions as

$$N_{\text{Train}} = round\{\propto . nV\ ar\} = round\{0.5 \times 4\} = 2\ , \quad retraining = \{2,10\}$$

$$def^{1}{}_{new} = att \times \left(1 - \sin\beta \times U(0,1)\right) + \frac{def_{old}+att}{2} \times \sin\beta \times U(0,1) \quad (81)$$

$$def^{2}{}_{new} = def_{old} \times \left(1 - \sin(\frac{\pi}{2} - \beta) \times U(0,1)\right) + \frac{def_{old} + att}{2} \times \sin(\frac{\pi}{2} - \beta) \times U(0,1) \quad (82)$$

Equations (81) and (82) illustrate the actions of two new defenders, based on the attacker and the defender, respectively. In both cases, the average distance between the attacker and the defense is the main region of movement. Furthermore, the attacker's present location in Equation (81) and the defender's current position in Equation (82) are the main driving forces for the defender's movement in this attack.

Similar to earlier locations, the movement is determined by the quantity of β needed to accomplish this goal and a random distribution *i.e.* U (0, 1).

3.3. *Diversion theft*

Using this tactic, the attacker initially leads the defense into a position that is actually a trick on the defender. This procedure is illustrated via the creation of a single solution. This action's formulation is displayed in the following equation:

$$def_{new} = def_{old} \times \left(1 - \sin\beta \times U(0,1)\right)\left(1 - \sin(\frac{\pi}{2} - \beta)\right) + \frac{(def_{old} + att \times U(0,1) \times \sin(\frac{\pi}{2} - \beta))}{2} \times \sin\beta \times U(0,1) \quad (83)$$

To this purpose, the defender's movement is presented by Eq. (74) which considers its current position as well as the average distance from a weighted number of attackers. In this spotting attack, the quantity of β and the uniform distribution U (0, 1) mostly influence the defender's movement.

3.4. *Pretext*

Using this tactic, the attacker steers the defender by baiting certain characteristics that the defender finds appealing. This method works well for taking out the defense. The following equation creates and generates a single solution during this process:

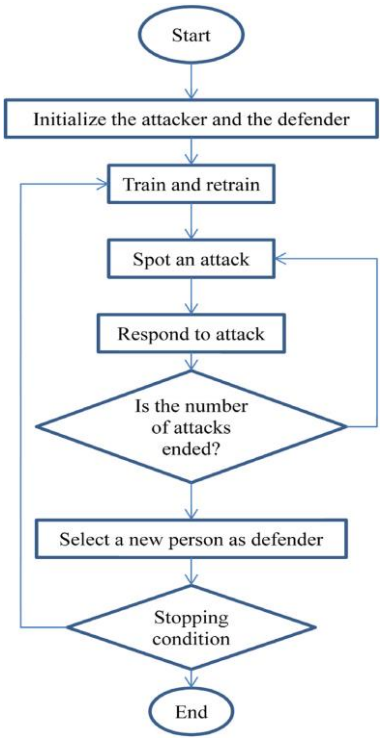
$$def_{new} = (def_{old} \times U(0,1) \times \sin\left(\frac{\pi}{2} - \beta\right))\left(1 - \sin\beta \times U(0,1)\right) + \frac{((def_{old} \times U(0,1) \times \sin(\frac{\pi}{2} - \beta)) + att)}{2} \times \sin\beta \times U(0,1) \quad (84)$$

There are two key terms in Eq. (84). The first one is a weighted calculation based on the defender's present location. The second phrase is the average distance between the attacker and the weighted defender. The pace at which the attack is detected and the uniform distribution between 0 and 1 also weigh the entirety of this distance.

Pseudo-code of SEO #9. [40]

```
T1=clock;
Initialize attacker and defender
It=1;
while solving_time < Max_time
    Do training and retraining;
    Num_attack=1;
    while Num_attack < Max_attack
        Spot an attack;
        Check the boundary;
        Respond to attack;
        if the OF of defender is lower than attacker
          Exchange the defender and attacker position;
        endif
        Num_attack= Num_attack+1;
    endwhile
    Create a new solution as defender;
    It=It+1;
    T2=clock;
    Solving_time=T2- T1;
endwhile
Return attacker.
```

Figure (9): The SEO flow chart. [40]



The advantages of SEO are:

1- Nature-inspired robustness and efficiency.
2- Effective global and local search strategies.
3- Preservation of solution diversity.
4- Adaptability to various problem domains.
5- Parallelizability for efficient computation.
6- Scalability for both small and large-scale problems.
7- Potential for hybridization with other techniques.
8- Ease of implementation and understanding.
9- Applicability to multi-objective optimization tasks

The limitations of SEO are:

1- Potentially slower convergence speed.
2- Sensitivity to parameter settings.
3- Dependence on population size.
4- Challenges in balancing exploration and exploitation.
5- Difficulty in handling complex constraints.
6- Adapting to dynamic environments can be challenging.
7- Varied performance across different problem types.
8- Difficulty in exploring multi-modal solution spaces.
9- High computational complexity for certain problems

Applications of SEO:

1- Solve a truck scheduling problem in a cross-docking system. [41]
2- Closed-Loop Supply Chain System with Uncertain Demand. [42]
3- Solving an Energy-Efficient Disassembly Line Balancing Problem Based on Bucket Brigades and Cloud Theory. [43]
4- A biobjective home health care logistics considering the working time and route balancing. [44]

7. HUMAN BEHAVIOR-BASED OPTIMIZATION (HBBO) [45]

The process of optimizing systems, procedures, and technologies based on human behavior and preferences is known as human behavior-based optimization, or HBBO. By considering how people interact with systems, HBBO seeks to improve their usability, intuitiveness, and efficiency.

The foundation of HBBO is the idea that people enjoy particular behaviors like efficiency, simplicity, and ease of use and will frequently repeat these behaviors when given the chance. HBBO aims to comprehend these behaviors and develop systems that are tailored to these preferences, making user experience more pleasurable and straightforward.

Applications for HBBO include product development, website design, and organizational procedures. Designers and developers can utilize data analytics and user research to find trends in user behavior and preferences, which they can then utilize to enhance the user experience in general.

In general, HBBO is a valuable strategy for developing user-friendly, efficient systems that can boost engagement, loyalty, and satisfaction among users.

The five steps of this algorithm are as follows:

Step 1: Setup
Step 2: Learning
Step 3: Talking with someone
Step 4: Probability of a field changing
Step 5: Completion

2.1 Setup

This process involves creating, evaluating, and distributing the initial personnel among the fields. An individual is defined as follows in an optimization issue using the variables $x_{N_{var}}$:

$$\text{Individual} = \left[x_1, x_2, \ldots., x_{N_{var}}\right] \quad (85)$$

After generating N_{pop} starting populations, the algorithm divides them equally among N_{field} initial fields. These people make up society.

The starting personnel in each field are as follows:

$$N.Ind_i = round\left\{\frac{N_{pop}}{N_{field}}\right\} \quad (86)$$

where $N.Ind_i$ is the number of original people in field i th. The function values of the initial individuals will be computed after they are generated. The definition of an individual's function value is as follows:

$$\text{Function value} = f\left(x_1, x_2, \ldots., x_{N_{var}}\right) \quad (87)$$

2.2 Learning

The algorithm will determine a radial coordinate (r) between $r_{min} = k_1 d$ and $r_{min} = k_1 d$ in an N-dimensional optimisation problem using the definition of spherical coordinate system for N-dimensional Euclidean space [21]. Here, d represents the Euclidian distance between the origin and individual, and k_i, an algorithm parameter, is the weighting factor.

Also, the process will determine N-1 random angular coordinates $(\theta_1, \theta_2, \ldots., \theta_{N-1})$, where the other angles will be chosen between 0 and π radians and θ_{N-1} will be found between 0 and 2π radians.

2.3 Talking with someone

These additional variables will replace the individual variables in this scenario. If there isn't a better function value for the new collection of variables, nothing will change. How many random variables will be changed is determined using the following method:

$$N_c = round\{\sigma \times N_{var}\} \quad (88)$$

The consultation factor, represented by the symbol, is an algorithmic parameter that determines the amount of random variables N_c that can be changed throughout the consultation process.

2.4 Probability of a field changing

This method, which is displayed below, orders each field according to its expert individual function value:

$$\text{Sort fields} = [field_1, field_2, \ldots., field_n] \quad (89)$$

The experts in $field_1$ and $field_2$ have the worst and best function values among the rest,

respectively. After that, the change probability for each field can be computed using the formula below:

$$P_i = \frac{O_i}{N_{field}+1} \quad (90)$$

where P_i and O_i stand for the i-th field's sort order and field-changing probability, respectively. The following equation is then checked by creating a random integer between 0 and 1, and if the expression is satisfied, the field changes for one of the people in this field:

if rand $\leq P_i \rightarrow$ field shifting takes place (91)

According to the function value, a selection probability for every individual will be defined in the field-changing operation as follows:

$$\mathrm{P.S_j} = \left| \frac{f(Individual_j)}{\sum_{k=1}^{N_{ind}} f(Individual_k)} \right| \quad (92)$$

where N_{ind} the total is number of people in the selected field and $\mathrm{P.S_j}$ is the selection probability for the j-th individual.

2.5 Completion

Dialogue and educational procedures cause people's positions to change. Thus, the function values of the individuals will be computed at this stage; if one of the halting conditions is met, the algorithm will terminate; if not, step 2 will be reached. The conditions for stopping are as follows:

(a) Iteration count has reached its maximum.
(b) As many function assessments as possible have been completed.
(c) When the average relative change in the objective function value over stall iterations is less, function tolerance is obtained.

HBBO's pseudo code #10. [45]

```
Initialize the parameters.
Create the starting population.
Population distribution across fields
as (i ≤ Maximum Iteration) do
    for each of the n fields (i = n), do
        for j = m (m number of people in field_i), do
            Education ( person_{i,j})
        end for
    end for
for i = n (the total number of people), do
        The consultation ( person_i)
    end for
    for each of the n fields (i = n), do
        for j = m (m number of people in field_i), do
            If the value of Individual_j in field_i = True, then
                Transfer Individual_j to an arbitrary field.
            end if
        end for
    end for
```

The benefits of HBBO include

1- This method shows that creative resource limitations do not exist for optimization and that surprising results can be achieved by carefully combining two seemingly unrelated scientific domains.
2- In terms of algorithm dependability, result correctness, and convergence time, HBBO outperforms other optimization algorithms, according to the experiment findings.
3- The results show that HBBO is the least CPU-intensive and fastest optimization method.
4- HBBO is an easy-to-use tool that solves a variety of difficult real-world optimization problems.

The HBBO's limitations are

1- Computational Complexity: HBBO algorithms can be computationally intensive, especially for large-scale or high-dimensional problems.
2- Parameter Sensitivity: They often require careful tuning of parameters, making them sensitive to settings and potentially requiring extensive experimentation.
3- Convergence to Suboptimal Solutions: HBBO algorithms may converge to suboptimal solutions or get stuck in local optima, particularly in complex fitness landscapes.
4- Limited Understanding of Behavior: Our incomplete understanding of human behavior may limit the effectiveness of HBBO algorithms that mimic it.
5- Over fitting and Generalization: HBBO algorithms may over fit to training data or environments, leading to poor generalization on new problems.
6- Lack of Domain Knowledge Incorporation: Integrating domain knowledge into HBBO algorithms can be challenging and may introduce biases or limitations.
7- Limited Scalability: Scaling HBBO algorithms to handle extremely large-scale problems or distributed environments may be difficult.

Applications of HBBO:

1- Application in many fields of science such as economics, business, computer science and aerospace, electrical, and mechanical engineering. [46].
2- Solving the Manufacturing Cell Design Problem [47].
3- Encouraged by Self-Organizing Maps to Address the S-Box Design Issue [48].
4- Reduced Reactive and Active Power Loss [49].

12. SEEKER OPTIMIZATION ALGORITHM (SOA) [50]

A metaheuristic optimization method called the Seeker Optimization method (SOA) was developed in response to the communal foraging habits of bees, ants, and birds. K.S. Lee and associates presented the algorithm in 2015.

Within the SOA, a group of "seekers "potential solutions travel across a search space in accordance with a set of guidelines inspired by the actions of foraging animals. The algorithm is made to adjust to changes in the environment (i.e., the fitness landscape) as the search progresses, balancing the exploration of the search space with the exploitation of promising regions.

These are the fundamental stages of the SOA:

1. Initialization: In the search space, create a random starting population of seekers.
2. Foraging: By randomly travelling with a specific probability, each seeker explores the search space. Depending on its fitness value, or how well the solution works, each seeker simultaneously draws other seekers to its position. This behavior is similar to how foraging animals exhibit social attraction.
3. Updating: Each seeker's position is modified based on its own movement and the attraction of other seekers after a predetermined number of iterations. Every seeker's fitness is also assessed and updated.
4. Termination: When a stopping requirement is satisfied, the search is terminated (e.g., a maximum number of iterations is reached, or a good solution is identified).

Numerous optimization issues, including as function optimization, feature selection, and clustering, have been addressed by the SOA. When compared to other metaheuristic algorithms like genetic and particle swarm optimization, it has demonstrated encouraging outcomes, particularly in high-dimensional and multimodal optimization issues.

Cloud Theory

The point at U where the cloud's center of gravity is located is called Ex . The measure of a concept's prevalence in conversation is called en. He is the entropy of the entropy En and measures the cloud droplet dispersion.

Given the three parameters (Ex, En, and He) of a normal cloud model, the following process, referred to as the basic normal cloud generator, generates the cloud with ncloud droplets [19].

Algorithm 1. A standard, basic cloud generator

Enter: Ex, En, He , n

Outcome: $\{(x_1, \mu_1), \ldots\ldots, (x_n, \mu_n)\}$

for i =1 to n

$En' = RANDN(En, He)$

$x_i = RANDN(Ex, En)$

$\mu_i = e^{\frac{-(x_i - Ex)^2}{2(En')^2}}$

$cloud(x_i, \mu_i)$

end.

In this case, a normally distributed random integer with mean a and standard deviation b is obtained using the function RANDN (a, b). The i th cloud drop in the cosmos is the $cloud(x_i, \mu_i)$.

Every seeker in the SOA has a start position vector $\vec{c}$, which may be thought of as the cloud model's expected value $Exthe$ and serves as the starting point for locating the next solution. Additionally, every seeker possesses a search direction $\vec{d}$ indicating his path, a trust degree $\vec{\mu}$ defined by the membership degree of the cloud model, and a search radius $\vec{r}$ equal to the En' of the cloud model.

The seeker goes to a new point $\vec{x}(t+1)$ at each time step t, where the four parameters are selected by search decision-making. A similar Y-conditional cloud generator [10] determines the update of the position from the start position by an uncertainty reasoning process that goes like this:

$$x_{ij}(t+1) = C_{ij} + d_{ij} r_{ij} (-\ln(\mu_{ij}))^{0.5} \quad (93)$$

where "j" is the index of variable dimensions and "i" is the index of seekers.

In this case, a normally distributed random integer with mean a and standard deviation b is obtained using the function RANDN (a, b). The i th cloud drop in the cosmos is the $cloud(x_i, \mu_i)$.

Every seeker in the SOA has a start position vector $\vec{c}$, which may be thought of as the cloud model's expected value $Exthe$ and serves as the starting point for locating the next solution. Additionally, every seeker possesses a search direction $\vec{d}$ indicating his path, a trust degree $\vec{\mu}$ defined by the membership degree of the cloud model, and a search radius $\vec{r}$ equal to the En' of the cloud model.

The seeker goes to a new point $\vec{x}(t+1)$ at each time step t, where the four parameters are selected by search decision-making. A similar Y-conditional cloud generator [10] determines the update of the position from the start position by an uncertainty reasoning process that goes like this:

$$x_{ij}(t+1) = C_{ij} + d_{ij} r_{ij} (-\ln(\mu_{ij}))^{0.5} \quad (93)$$

where "j" is the index of variable dimensions and "i" is the index of seekers.

The primary algorithm's pseudo code. #11. [50]

```
begin
  t•0;
  generating S positions randomly and
   uniformly;
  repeat
  evaluating each seeker;
  giving search parameters: start position,
   search
    direction, search radius, and trust degree;
  updating positions using (1);
  t•t+1;
  until t = T_max
end.
```

Parameters of Algorithms

1- Vector at the Start Point

2- The current position $\vec{x}(t)$ is intuitively assigned as the initial position vector$\vec{c}$. Motivated by PSO, each seeker has a memory that stores both its current best position $\vec{p}$ and the global best position $\vec{g}$ that it has learned from interacting with other nearby seekers.

$$\vec{c} = \vec{x}(t) + \emptyset_2\left(\vec{g}(t) - \vec{x}(t)\right) + \emptyset_1\left(\vec{p}(t) - \vec{x}(t)\right). \quad (94)$$

In the interval [0, 1], real numbers $\emptyset_1$ and $\emptyset_2$ are randomly selected and uniformly chosen.

3- Direction of Search

Two local temporal directions d_{lt}, one local special direction d_{ls}, one global temporal direction d_{gt}, and one global special direction d_{gs} are associated with each seeker.

$$\vec{d}_{lt} = \begin{cases} sign\left(\vec{x}(t) - \vec{x}(t-1)\right) \ if \ fit\left(\vec{x}(t)\right) \geq fit\left(\vec{x}(t-1)\right) \\ sign\left(\vec{x}(t-1) - \vec{x}(t)\right) \ if \ fit\left(\vec{x}(t)\right) < fit\left(\vec{x}(t-1)\right) \end{cases} \quad (95)$$

$$\vec{d}_{ls} = sign\left(\vec{x}'(t) - \vec{x}(t)\right) \quad (96)$$

$$\vec{d}_{gt} = sign\left(\vec{p}(t) - \vec{x}(t)\right) \quad (97)$$

$$\vec{d}_{gs} = sign\left(\vec{g}(t) - \vec{x}(t)\right) \quad (98)$$

Fit $(\vec{x}(t))$ is the fitness function of $\vec{x}(t)$, where sign (•) is a signum function, and $\vec{x}'(t)$ is the location of the seeker with the greatest fitness in a particular neighborhood region.

After that, we can designate the search direction in the following ways based on the four directions.

$$\vec{d} = sign\left(\omega\left(\vec{x}(t) - \vec{x}(t-1)\left(sign\left(fit\left(\vec{x}(t)\right) - fit\left(\vec{x}(t-1)\right)\right)\right) + \emptyset_2\left(\vec{g}(t) - \vec{x}(t)\right) + \emptyset_1\left(\vec{p}(t) - \vec{x}(t)\right)\right) \quad (99)$$

where $\omega = (T_{max} - t)/T_{max}$ and $\emptyset_1$ and $\emptyset_2$ are uniformly and randomly generated real values inside a defined interval [0, 1].

1- Radius of Search

It is necessary, but difficult, to give a reasonable search radius.

Algorithm 2. The cloud based method of the search radius

$$\mathrm{En_r} = \vec{x}_{\max} - \vec{x}_{\min}$$
$$\mathrm{He_r} = \frac{\mathrm{En_r}}{10};$$
$$\vec{r'} = \mathrm{RANDN}\ (\mathrm{En_r}, \mathrm{He_r});$$
$$\vec{r} = \mathrm{RANDN}\left(0, \vec{r'}\right);$$

Where, within its fellow neighbor, $\vec{x}_{\max}$ and $\vec{x}_{\min}$ denote the places with the maximum and minimum fitness, respectively. For example, the issue domain's "known" region could be considered to be the En. The real numbers picked consistently and at random inside the provided interval $\left(0, \vec{r'}\right)$ constitute the function RANDN $\left(0, \vec{r'}\right)$.

In order to reduce computing time, the search radius was calculated using the simple approach $\vec{r} = \mathrm{RANDN}\ (0, \mathrm{En_r})$, where ALGORITHM 2 is used for $\mathrm{En_r}$.

In other words, reasoning about uncertainty was handled using fuzzy logic.

2- Degree of Trust

Actually, the fuzzy set theory and cloud model's grade of membership is the parameter μ. The uncertainty rule of intelligent search can be defined as follows: "If {fitness is large}, then {search radius is small}," as discussed in section 1.For "big" levels of "fitness," a linear membership function was employed. To be more precise, it is either directly proportional to the fitness $\vec{x}$ or, in the case of our trials, the index of the ascensive sort order of the fitness of $\vec{x}$. To put it another way, the best position currently has the greatest μ_{max}, the other location has a μ<1.0, and the worst position currently has the least μ_{min}. The expression is shown as (8) and (9).

$$\mu_i = \mu_{\max} - \frac{\mathrm{S} - \mathrm{I_i}}{\mathrm{S} - 1}\left(\mu_{\max} - \mu_{\min}\right). \quad (100)$$

$$\mu_{i,j} = \mathrm{RAND}\left(\mu_i, 1\right) \quad (101)$$

Where S is the neighbor search group size, and $\mathrm{I_i}$ is the index (sequence number) of $\vec{x}_i$ after sorting the finesses of neighbor seekers in ascending order.

SOA has the following benefits:

1- Because cloud theory preserves uncertainty in transition, it effectively meets the criteria of real-life settings.

2- Has been effectively used in intelligent control (9) and data mining (10).
3- Much different from the current search techniques.
4- Found the global optimum faster, stronger, and more efficiently than with PSO and GA.
5- Solved exceptionally well, converging to almost global optimal solutions across multiple classes of problems with different degrees of complexity.

The following are SOA's limitation:

1- Although it takes its cues from fuzzy logic theory, the cloud theory (8) tackles the drawbacks of overly precise and rigorous specifications.
2- It appears in commonly used transition models and interferes with the process of human recognition.

SOA's applications:

1- Optimal reactive power dispatch on standard IEEE 57- and 118-bus power systems. [51]
2- Design of digital filters with infinite-impulse-response (IIR). [52]
3- Voltage stability in reactive power dispatch. [53]
4- Neural network structure and parameter tuning. [54]
5- Solving economic dispatch issues. [55]
6- A unique global numerical optimization technique using stochastic search. [56]

13. CONCLUSION

Human-Based Optimization Algorithms (HBOAs) offer a flexible and adaptable approach to optimization, inspired by human problem-solving strategies. They can be tailored to specific problems, integrated with other techniques, and applied across various real-world domains. Despite facing challenges such as scalability and parameter tuning, HBOAs show promise for driving innovation and addressing complex optimization tasks in diverse fields. Ongoing research aims to refine these algorithms and explore new applications, underscoring their potential significance in advancing optimization methodologies. The benefits, drawbacks, and uses of human-based meta-heuristic algorithms are discussed. The purpose of the paper is to provide a quick overview of many human-based meta-heuristic methods for optimization problem solving. This research thoughtfully discusses eleven human-based algorithms.

Future Scope of work is Advancements in Artificial Intelligence and Machine Learning, Applications in Multi-objective Optimization, Integration with Big Data, Hybridization, Real-Time Applications,

Human-based optimization algorithms have a promising future, particularly in addressing complex, real-world problems requiring innovative, adaptive, and human-like decision-making capabilities.

REFRENCES

[1] Dehghani, Mohammad, Mohammad Mardaneh, Josep M. Guerrero, Om Parkash Malik, Ricardo A. Ramirez-Mendoza, José Matas, Juan C. Vasquez, and Lizeth Parra-Arroyo. "A new "Doctor and Patient" optimization algorithm: An application to energy commitment problem." Applied Sciences 10, no. 17 (2020): 5791.

[2] Dehghani, Mohammad, and Pavel Trojovský. "Teamwork optimization algorithm: A new optimization approach for function minimization/maximization." Sensors 21, no. 13 (2021): 4567.

[3] Mousavirad, Seyed Jalaleddin, and Hossein Ebrahimpour-Komleh. "Human mental search: a new population-based metaheuristic optimization algorithm." Applied Intelligence 47 (2017): 850-887.

[4] Moosavi, Seyyed Hamid Samareh, and Vahid Khatibi Bardsiri. "Poor and rich optimization algorithm: A new human-based and multi populations algorithm." Engineering applications of artificial intelligence 86 (2019): 165-181.

[5] Dehghani, Mohammad, Zeinab Montazeri, Ali Dehghani, Ricardo A. Ramirez-Mendoza, Haidar Samet, Josep M. Guerrero, and Gaurav Dhiman. "MLO: Multi Leader Optimizer." International Journal of Intelligent Engineering & Systems 13, no. 6 (2020).

[6] Dehghani, M., M. Mardaneh, and O. P. Malik. "FOA: 'Following'Optimization Algorithm for solving Power engineering optimization problems." Journal of Operation and Automation in Power Engineering 8, no. 1 (2020): 57-64.

[7] Klau, Gunnar W., Neal Lesh, Joe Marks, and Michael Mitzenmacher. "Human-guided tabu search." In AAAI/IAAI, pp. 41-47. 2002.

[8] Dehghani, Mohammad, Eva Trojovská, and Pavel Trojovský. "A new human-based metaheuristic algorithm for solving optimization problems on the base of simulation of driving training process." Scientific reports 12, no. 1 (2022): 9924.

[9] Kannan, B. K., and Steven N. Kramer. "An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design." (1994): 405-411.

[10] Mirjalili, Seyedali, and Andrew Lewis. "The whale optimization algorithm." Advances in engineering software 95 (2016): 51-67.

[11] Trojovská, Eva, and Mohammad Dehghani. "A new human-based metahurestic optimization method based on mimicking cooking training." Scientific Reports 12, no. 1 (2022): 14861.

[12] Mirjalili, Seyedali, and Andrew Lewis. "The whale optimization algorithm." Advances in engineering software 95 (2016): 51-67.
[13] Koziel, Slawomir, and Xin-She Yang, eds. Computational optimization, methods and algorithms. Vol. 356. Springer, 2011.
[14] Mezura-Montes, Efrén, and Carlos A. Coello Coello. "Useful infeasible solutions in engineering optimization with evolutionary algorithms." In Mexican international conference on artificial intelligence, pp. 652-662. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.
[15] Rao, R. Venkata, Vimal J. Savsani, and Dipakkumar P. Vakharia. "Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems." Computer-aided design 43, no. 3 (2011): 303-315.
[16] Dastan, Mohammadhossein, Saeed Shojaee, Saleh Hamzehei-Javaran, and Vahid Goodarzimehr. "Hybrid teaching–learning-based optimization for solving engineering and mathematical problems." Journal of the Brazilian Society of Mechanical Sciences and Engineering 44, no. 9 (2022): 431.
[17] Tiwari, Atul, and M. K. Pradhan. "Applications of TLBO algorithm on various manufacturing processes: AReview." Materials Today: Proceedings 4, no. 2 (2017): 1644-1652.
[18] Rao, R. Venkata, and Vivek Patel. "An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems." Scientia Iranica 20, no. 3 (2013): 710-720.
[19] Dede, Tayfun. "Application of teaching-learning-based-optimization algorithm for the discrete optimization of truss structures." Ksce journal of civil engineering 18 (2014): 1759-1767.
[20] Xiaowen, Zhao, Yang Qingshan, and Zhang Yunhua. "Application of TLBO to synthesis of sparse concentric ring arrays." In 2016 10th European Conference on Antennas and Propagation (EuCAP), pp. 1-5. IEEE, 2016.
[21] Sulaiman, Mohd Herwan, Zuriani Mustaffa, and Muhammad Ikram Mohd Rashid. "An application of teaching–learning-based optimization for solving the optimal power flow problem with stochastic wind and solar power generators." Results in Control and Optimization 10 (2023): 100187.
[22] Hubalovska, Marie, and Stepan Major. "A New Human-Based Metaheuristic Algorithm for Solving Optimization Problems Based on Technical and Vocational Education and Training." Biomimetics 8, no. 6 (2023): 508.
[23] Dehghani, Mohammad, Eva Trojovská, and Tomáš Zuščák. "A new human-inspired metaheuristic algorithm for solving optimization problems based on mimicking sewing training." Scientific Reports 12, no. 1 (2022): 17387.
[24] Moghdani, Reza, and Khodakaram Salimifard. "Volleyball premier league algorithm." Applied Soft Computing 64 (2018): 161-185.
[25] Moghdani, Reza, Mohamed Abd Elaziz, Davood Mohammadi, and Nabil Neggaz. "An improved volleyball premier league algorithm based on sine cosine algorithm for global optimization problem." Engineering with Computers 37 (2021): 2633-2662.
[26] Abd Elaziz, Mohamed, Neggaz Nabil, Reza Moghdani, Ahmed A. Ewees, Erik Cuevas, and Songfeng Lu. "Multilevel thresholding image segmentation based on improved volleyball premier league algorithm using whale optimization algorithm." Multimedia Tools and Applications 80 (2021): 12435-12468.
[27] Salimifard, Khodakaram, Jingpeng Li, Davood Mohammadi, and Reza Moghdani. "A multi objective volleyball premier league algorithm for green scheduling identical parallel machines with splitting jobs." Applied Intelligence 51, no. 7 (2021): 4143-4161.
[28] Sun, Shuo, Liang Ma, and Yong Liu. "A competitive Volleyball Algorithm to solve global optimization with the engineering problems." (2021).
[29] Trojovský, Pavel, and Mohammad Dehghani. "A new optimization algorithm based on mimicking the voting process for leader selection." PeerJ Computer Science 8 (2022): e976.
[30] Abd Elaziz, Mohamed, Mohamed E. Zayed, H. Abdelfattah, Ahmad O. Aseeri, Elsayed M. Tag-eldin, Manabu Fujii, and Ammar H. Elsheikh. "Machine learning-aided modeling for predicting freshwater production of a membrane desalination system: A long-short-term memory coupled with election-based optimizer." Alexandria Engineering Journal 86 (2024): 690-703.
[31] Pandiyaraju, V., Sannasi Ganapathy, N. Mohith, and A. Kannan. "An optimal energy utilization model for precision agriculture in WSNs using multi-objective clustering and deep learning." Journal of King Saud University-Computer and Information Sciences 35, no. 10 (2023): 101803.
[32] Kumar, B. Naresh, and Jai Sukh Paul Singh. "Efficient Optimization Algorithm for IoT-based Cognitive Radio Routing Model with Energy Harvesting for Medical Data Transmission." In *2023 International Conference on the Confluence of Advancements in Robotics, Vision and Interdisciplinary Technology Management (IC-RVITM)*, pp. 1-9. IEEE, 2023.
[33] Rampriya, R., N. Suguna, RG Suresh Kumar, and P. Sudhakar. "Optimal Feature Selection and Classification of Respiratory Diseases by Novel EFICNN-EBOA Algorithm: A Real Time Implementation Concept." International Journal of Intelligent Systems and Applications in Engineering 12, no. 2s (2024): 699-712.

[34] Arora, Sankalap, Manik Sharma, and Priyanka Anand. "A novel chaotic interior search algorithm for global optimization and feature selection." Applied Artificial Intelligence 34, no. 4 (2020): 292-328.
[35] Moravej, Mojtaba, and Seyed-Mohammad Hosseini-Moghari. "Large scale reservoirs system operation optimization: the interior search algorithm (ISA) approach." Water Resources Management 30 (2016): 3389-3407.
[36] Gandomi, Amir H., and David A. Roke. "Engineering optimization using interior search algorithm." In 2014 IEEE symposium on swarm intelligence, pp. 1-7. IEEE, 2014.
[37] El-Hay, E. A., M. A. El-Hameed, and A. A. El-Fergany. "Optimized parameters of SOFC for steady state and transient simulations using interior search algorithm." Energy 166 (2019): 451-461.
[38] Han, Bo, Changqiang Huang, Shangqin Tang, Yongbo Xuan, Zhuoran Zhang, and Zhou Huan. "Random orthocenter strategy in interior search algorithm and its engineering application." Soft Computing 24, no. 8 (2020): 5933-5948.
[39] Chandrasekaran, Shilaja. "Multiobjective optimal power flow using interior search algorithm: A case study on a real-time electrical network." Computational Intelligence 36, no. 3 (2020): 1078-1096.
[40] Fathollahi-Fard, Amir Mohammad, Mostafa Hajiaghaei-Keshteli, and Reza Tavakkoli-Moghaddam. "The social engineering optimizer (SEO)." Engineering applications of artificial intelligence 72 (2018): 267-293.
[41] Fathollahi-Fard, Amir Mohammad, Mehdi Ranjbar-Bourani, Naoufel Cheikhrouhou, and Mostafa Hajiaghaei-Keshteli. "Novel modifications of social engineering optimizer to solve a truck scheduling problem in a cross-docking system." Computers & Industrial Engineering 137 (2019): 106103.
[42] Aghamohamadi, Soroush, Masoud Rabbani, and Reza Tavakkoli-Moghaddam. "A social engineering optimizer algorithm for a closed-loop supply chain system with uncertain demand." International Journal of Transportation Engineering 9, no. 1 (2021): 521-536.
[43] Tian, Guangdong, Cheng Zhang, Amir M. Fathollahi-Fard, Zhiwu Li, Chaoyong Zhang, and Zhigang Jiang. "An enhanced social engineering optimizer for solving an energy-efficient disassembly line balancing problem based on bucket brigades and cloud theory." IEEE Transactions on Industrial Informatics (2022).
[44] Goodarzian, Fariba, Ajith Abraham, and Amir Mohammad Fathollahi-Fard. "A biobjective home health care logistics considering the working time and route balancing: a self-adaptive social engineering optimizer." Journal of Computational Design and Engineering 8, no. 1 (2021): 452-474.
[45] Soto, Ricardo, Broderick Crawford, Francisco González, Emanuel Vega, Carlos Castro, and Fernando Paredes. "Solving the manufacturing cell design problem using human behavior-based algorithm supported by autonomous search." IEEE Access 7 (2019): 132228-132239.
[46] Ahmadi, Seyed-Alireza. "Human behavior-based optimization: a novel metaheuristic approach to solve complex optimization problems." Neural Computing and Applications 28, no. Suppl 1 (2017): 233-244.
[47] Soto, Ricardo, Broderick Crawford, Francisco González Molina, and Rodrigo Olivares. "Human behaviour based optimization supported with self-organizing maps for solving the S-box design Problem." IEEE Access 9 (2021): 84605-84618.
[48] Soto, Ricardo, Broderick Crawford, Francisco González Molina, and Rodrigo Olivares. "Human behaviour based optimization supported with self-organizing maps for solving the S-box design Problem." IEEE Access 9 (2021): 84605-84618.
[49] Parmar, Siddharth, Indrajit Trivedi, R. H. Bhesdadiya, and Pradeep Jangir. "Reactive and Active Power Losses Minimization using Human Behavior Based Optimization." (2016).
[50] Dai, Chaohua, Yunfang Zhu, and Weirong Chen. "Seeker optimization algorithm." In Computational Intelligence and Security: International Conference, CIS 2006. Guangzhou, China, November 3-6, 2006. Revised Selected Papers, pp. 167-176. Springer Berlin Heidelberg, 2007.
[51] Dai, Chaohua, Weirong Chen, Yunfang Zhu, and Xuexia Zhang. "Seeker optimization algorithm for optimal reactive power dispatch." IEEE Transactions on power systems 24, no. 3 (2009): 1218-1231.
[52] Dai, Chaohua, Weirong Chen, and Yunfang Zhu. "Seeker optimization algorithm for digital IIR filter design." IEEE transactions on industrial electronics 57, no. 5 (2009): 1710-1718.
[53] Dai, Chaohua, Weirong Chen, Yunfang Zhu, and Xuexia Zhang. "Reactive power dispatch considering voltage stability with seeker optimization algorithm." Electric Power Systems Research 79, no. 10 (2009): 1462-1471.
[54] Dai, Chaohua, Weirong Chen, Yunfang Zhu, Zhiling Jiang, and Zhiyu You. "Seeker optimization algorithm for tuning the structure and parameters of neural networks." Neurocomputing 74, no. 6 (2011): 876-883.
[55] Shaw, Binod, V. Mukherjee, and S. P. Ghoshal. "Solution of economic dispatch problems by seeker optimization algorithm." Expert Systems with Applications 39, no. 1 (2012): 508-519.
[56] Dai, Chaohua, Weirong Chen, Yonghua Song, and Yunfang Zhu. "Seeker optimization algorithm: a novel stochastic search algorithm for global numerical optimization." Journal of Systems

Engineering and Electronics 21, no. 2 (2010): 300-311.
[57] Rao, R. V., & Patel, V. (2012). Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design.
[58] Mirjalili, S., & Lewis, A. (2016). The Whale Optimization Algorithm. *Advances in Engineering Software*.
[59] Dutta, S., et al. (2020). Applications of human-based optimization techniques in healthcare: A review. *Expert Systems with Applications*.