STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **15**, June 2026, pp 5022–5042.
Published online in International Academic Press (www.IAPress.org)

Improving Pelican Optimization Algorithm for Solving Integer and Mixed Integer Optimization Problems

Lamyaa Jasim Mohammed [1], Ghalya Tawfeeq Basheer [1], Zakariya Yahya Algamal [2,*]

[1]*Department of Operations Research and Intelligent Techniques, University of Mosul, Mosul, Iraq*
[2]*Department of Statistics and Informatics, University of Mosul, Mosul, Iraq*

Abstract Mixed-integer linear programming (MILP) problems, prevalent in logistics, scheduling, and resource allocation, pose significant computational challenges due to their NP-hard nature, prompting the need for efficient metaheuristic approaches. This study proposes an enhanced pelican optimization algorithm (POA) that adapts the original population-based method inspired by pelican hunting strategies of prey approach (exploration) and surface winging (exploitation) for discrete domains via novel tent-shaped transfer functions. These functions enable seamless discretization of continuous solutions into binary and integer variables, offering computational simplicity and gradual curvature superior to traditional S- and V-shaped transfers. Tent-shaped functions outperformed others, yielding lowest standard deviation values and means closest to optima, demonstrating superior stability and precision for practical optimization.

Keywords linear programming, mixed integer problem, pelican optimization algorithm, transfer function

DOI: 10.19139/soic-2310-5070-3500

1. Introduction

The optimization problems in the real world are usually very difficult to solve and most of the applications must contend with problems. Optimization tools must be applied to address such issues although it is not guaranteed that the optimal solution may be found [1]. Also, new algorithms have been designed to determine whether they can handle such difficult optimization problems. Some of these new algorithms include particle algorithms. Swarm optimization, cuckoo search and firefly algorithm have become popular because they are very high efficiency [2, 3, 4]. In this article, we have applied pelican algorithm the following general form can be expressed a problem as:

$$\text{Max } \mathrm{c}^{\mathrm{T}}\mathrm{x} \tag{1}$$

$$\text{s.t. } \mathrm{Ax} \leq \mathrm{b} \tag{2}$$

$$\mathrm{x} \in \mathrm{Z}^{\mathrm{n}} \tag{3}$$

where the solution $\mathrm{x} \in \mathrm{Z}^{\mathrm{n}}$ is a vector of n integer variables:$\mathrm{x} = (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_{\mathrm{n}})^{\mathrm{T}}$ and the data are rational and are given by the $\mathrm{m}\times\mathrm{n}$ matrix A, the $1\times\mathrm{n}$ matrix c , and the $\mathrm{m}\times 1$ matrix b . This formulation includes also equality constraint, because each equality constraint can be represented by means of two inequality constraints like those included in Eq. (2).

*Correspondence to: Zakariya Yahya Algamal (Email: zakariya.algamal@uomosul.edu.iq). Department of Statistics and Informatics, University of Mosul, Mosul, Iraq.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

A large number of real-life optimization problems (transport problems, problems of scheduling and distribution, resources planning, etc.) are offered by linear mixed integer models. The integer variables are specific items, which cannot be divided the continuous variables reflect (number of machines, vehicles and others).

Nowadays Mixed the application of Integer Problems (MIP) is decision oriented in most industrial and business operations. This is not only because of the increased both to the computing power of contemporary computers, and to better MIP solvers. facilitate simpler construction and solving of such models [5, 6, 7, 8].

The integer problems which have not continuous computing complexity the combination nature of the variables is the source of their combination nature. In mixed integer problems the computing complexity higher, as it is required to determine both the optimal sum of the integer variables and the optimum value of the continuous variables. The issues of this kind are part of the Non-deterministic Polynomial-time (NP) class. The famous ways of optimal solution finding (like branch and-bound or cutting planes) are exponentially complex in computing need a lot of computing time even on not large problems [9, 10]. These The existence of a large number of heuristics has been caused by computing problems algorithms. Heuristics play a very significant role in applied optimization [11, 12]. They can only be applied in the solution of large and complex problems, which are met very often practice. Their primary aim is to guarantee reasonable time of solutions of the optimization problem, near optimal feasible solution. In addition, they are able to be very helpful in complicated accurate procedures of determining possible initial solutions as well as to accelerate the search procedure at specified stages of such techniques (e.g. in a node of the branch and bound tree) [13, 14, 15]. The heuristic algorithms can be separated into two categories: problem-oriented, having a particular structure [16, 17] and general purpose techniques . To be applicable, it must be used in order to find application in the heuristic algorithms are tested against commercial software products [10]. their effectiveness, and to the universality of techniques, i.e., to their applicability and success on various types of problems [8, 11, 18].

2. Mixed Integer Programming

The term linear integer programming is used to refer to the family of combination constrained optimization problems whose variables are integer, whose objective is a linear function and whose constraints are linear inequalities. Linear Integer Programming (or LIP) optimization Integer programming is used to solve the problem posed by the non-integer solutions in the cases where integer values are needed. rounding might influence the profit or the cost by millions of dollars. Here we must have the problem solved in such a way that an optimal integer solution is ensured. Integer programming: as a pure integer linear programming, in which all the variables are integer-valued, it provides the possibility to find integer values. has to be integer, or as a mixed integer linear programming that permits some that the variables are continuous, or a 0-1 integer model, the decision variables are integer valued with zero or one. Many real-life problems in logistics, economics, social sciences and politics can be modeled as linear integer optimization problems.Other combination problems as the knapsack-capital budgeting problem, warehouse location problem, travelling salesman problem, decreasing costs and machinery selection problem, network and graph problems, such as maximum flow problems, set covering problems, matching problems, weighted matching problems, spanning tree problems and many scheduling problems can also be formulated as linear integer optimization problems. Cutting plane techniques are exact integer programming techniques [19, 20].

The branch and the bound are both computationally expensive, in large scale problems [19, 20]. The branch and the bound algorithms are very superior to the algorithms that utilize cutting planes alone. An example of such benefits is that the algorithms can be terminated prematurely provided at least one integral solution has been identified and a solution can be provided which is achievable even though it may not be optimal. In addition, the LP relaxations provide a worst-case estimate of the distance to optimal of the returned solution using their solutions. Lastly, multiple optimal solutions can be returned by the branch method and the bound method [2].

The integer linear programming is NP complete, hence due to this reason a large number of problems are intractable. Therefore, the heuristic methods have to be employed instead of the integer linear programming. As an example, Swarm intelligence meta heuristics, one of which is an ant colony optimization, artificial bee colony

optimization particle swarm optimization .To solve integer programming problems, most of the heuristics truncate or round the real valued solutions to the nearest integer values [2, 19, 20].

Mixed Integer Linear Programs (MILPs) are mathematical optimization problems that have two types of variables, discrete and continuous variables. The objective is also linear as well as the constraints. This class of optimization problems arises in numerous real-life problems across fields. In fact, MILP models can be used to formulate many real problems, such as: packing, knapsack, inventory, production planning, location, resource allocation, routing and scheduling problems, just to mention but a few [21]. Such a wide applicability has made the development of efficient algorithms to solve this general and popular type of optimization problems even more interesting. MILP problems are NP-hard. The impossibility to solve a great variety of important optimization problems has led to the development of approximation algorithms [22].

It is very common to be faced with the situation that the problem is NP-hard and it is really hard to find an optimal solution within a reasonable amount of time [23]. In general solving MILPS depend on two from of algorithms: exact algorithms and heuristics, to all intents and purposes. (e.g. branch and bound, branch and cut, branch and price) tend to be generally applicable, but have been found to be tedious on large or more complicated problems. Heuristics and especially meta heuristics can be employed when the instances are too large or challenging to solve precisely. A meta heuristic is an overarching process to choose between various heuristics. It is possible to consider two categories of meta heuristics: single solution algorithms (e.g. local search, tabu search) and population-based algorithms (e.g. evolutionary algorithms, swarm optimization) [22, 24, 25, 26].

We consider a generic mixed integer linear program (MILP) in the form

$$\min \mathrm{c}^{\mathrm{T}}\mathrm{x} \tag{4}$$

$$\mathrm{Ax} \geq \mathrm{b} \tag{5}$$

$$\mathrm{x_j} \in \{0\ ,\ 1\} \qquad \forall \quad \mathrm{j} \in \mathrm{B}, \tag{6}$$

$$\mathrm{x_j}\ \text{integer}\ \forall \qquad \mathrm{j} \in \mathrm{G} \tag{7}$$

$$\mathrm{x_j} \quad \text{continuous}\ \ \forall\, \mathrm{j} \in \mathrm{C} \tag{8}$$

where A is an $\mathrm{m}\times\mathrm{n}$ input matrix, b and c are input vectors of dimension m and n, respectively. Here, the variable index set $\mathrm{N} := \{1, ..., \mathrm{n}\}$ is partitioned into $(\mathrm{B}, \mathrm{G}, \mathrm{C})$, where B is the index set of the $0-1$variables (if any), while the sets G and C index the general integer and the continuous variables, respectively. Bound on the variables, including the $0-1$ bounds on the binary ones, are assumed to be part of system Eq. (5). Removing the integrity requirement on variables indexed by $\mathrm{I} := \mathrm{B}\cup\mathrm{G}$ leads to the LP relaxation $\min\ \{\mathrm{c}^{\mathrm{T}}\mathrm{x}\ :\mathrm{x} \in \mathrm{P}\ \}$where $\mathrm{P}\coloneqq\{\mathrm{x} \in \mathrm{R}^{\mathrm{n}}:\ \mathrm{Ax} \geq \mathrm{b}\}$. MILP heuristics aim at finding a feasible (and hopefully good) solution of the problem above, which is an NP-hard problem by itself [22].

3. Literature Review

A large number of practical optimization problems have decision variables whose values must be integer. Such problems are numerous in applications; e.g. plant operation, design, location, scheduling and set covering, set partitioning, set packing problems and allocation models [27]. The previous approaches to IP problem solutions belong to [28] and to [29]. These algorithms are aimed at solving restricted sub classes of IP problems and not the general type of IPPs. The overall IP issues in certain cases may be, approximated and solved as a linear programming (LP) problem and the solution to the LP optimization problem rounded off. Such a strategy can offer a viable solution, yet the solution might not be close to the best one. To solve integer linear programming problems (ILP) many methods have been proposed such as branch and bound (B&B), cutting planes, polyhedral

developments, hybrid algorithms, reformulation- linearization technique (RLT), facial disjunctive programming and post-solution analysis [30, 46, 47, 48].

The optimization of reliability problems with separable objective function and numerous constraints have also done much work. [31] give such examples. The majority of them will not give a global optimum answer. An approach called 0-1 integer nonlinear programming [32] is the optimization technique developed to solve integer nonlinear programming problems (INLP) variables. Their approach is a near relation to the lexicographic approach to the knapsack problem by [33] , and the additive algorithm, introduced by [34]. [32] presuppose the objective function to be a monotone non decreasing one and suppose that the functional constraints may be written (rewritten) in the form of a difference between two functions that happen to be monotone non decreasing. [35] came up with an algorithm to solve a spare allocation problem that can be used in the integer problem solved using a non-0-1 variable. In their approach, they termed it New Lawler and Bell. Generalization Using [32] algorithm, [36] generalized the Lawler and Bell algorithm, which solves a larger class of INLP problems. [37] provided two phase based optimization operations to integer programming problems. [38] introduced a survey on the non-standard methods of the integer programming problems.

A number of computational methods (including B&B, cutting planes, relaxation, outer approximation etc.) that are reasonably effective when used on many problems have been suggested in the literature on ILPs. Nevertheless ,no computational algorithm has been developed so far that could say that it can effectively solve a large number of INLPs [39].

4. Pelican Optimization Algorithm (POA)

The enormous pelican uses a big bag in its throat to catch and consume prey. It also has a long beak. This bird, which inhabits groups of several hundred pelicans, enjoys social interactions and group living. Pelicans have the following physical characteristics: they are between 2.75 and 15 kg in weight, 1.06 and 1.83 m in height, and 0.5 and 3 m in wingspan [40, 41]. Fish is a pelican's primary dietary source, with frogs, turtles, and crabs appearing less frequently [42]. When extremely hungry, pelicans may even consume seafood. Pelicans frequently cooperate during hunting. Once they have located their meal, the pelicans dive from a height of 10 to 20 meters to reach it. Lower altitudes are where certain creatures hunt [43]. They then extend their wings over the water to deceive fish into areas of shallower water, which makes it easier for them to catch and consume the fish. The pelican must bend its head forward before swallowing to prevent water from getting into its beak as it catches fish. Pelicans' astute hunting behaviors and strategies have allowed them to develop into expert hunters. The main source of inspiration for the design of POA [44, 45]. of course, some species also fly lower to reach their food. The fish are then forced to move to shallow water by their spreading wings on the water's surface, which makes it easier for them to catch fish. When a pelican catches a fish, a lot of water gets into its beak; to get rid of the extra water, it tilts its head forward and swallows the fish [51, 49, 50]. The cognitive process of pelican behavior and hunting strategy has made these birds adept hunters. The previously indicated strategy's modeling served as the primary source of inspiration for the design of POA [41, 53, 54, 55].

Using a population-based approach, the proposed POA counts pelicans as members of its population. Every person in a population-based algorithm represents a potential solution. Each population member suggests parameter values for the optimization problem based on their search space position. We first use Eq. (9) to randomly initialize population members based on the problem's upper and lower bounds.

$$x_{i,j} = l_j + \text{rand}.\,(u_j - l_j)\,, \quad i = 1, 2, \ldots, N\,, \qquad j = 1, 2, \ldots, m \tag{9}$$

where $x_{i,j}$ is the value of the jth variable specified by the ith candidate solution, N is the number of population members, m is the number of problem variables, rand is a random number in interval $[0, 1]$, l_j is the jth lower bound, and u_j is the jth upper bound of problem variables.

The population members of pelicans in the proposed POA are identified using a matrix called the population matrix in Eq. (10). Each row of this matrix represents a candidate solution, while the columns of this matrix

represent the proposed values for the problem variables.

$$\mathrm{X}=\begin{bmatrix}\mathrm{X}_1\\ \vdots\\ \mathrm{X}_\mathrm{i}\\ \vdots\\ \mathrm{X}_\mathrm{N}\end{bmatrix}_{\mathrm{N}\times\mathrm{m}}=\begin{bmatrix}\mathrm{x}_{1,1} & \cdots & \mathrm{x}_{1,\mathrm{j}} & \cdots & \mathrm{x}_{1,\mathrm{m}}\\ \vdots & \ddots & \vdots & \vdots & \vdots\\ \mathrm{x}_{\mathrm{i},1} & \cdots & \mathrm{x}_{\mathrm{i},\mathrm{j}} & \cdots & \mathrm{x}_{\mathrm{i},\mathrm{m}}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \mathrm{x}_{\mathrm{N},1} & \cdots & \mathrm{x}_{\mathrm{N},\mathrm{j}} & \cdots & \mathrm{x}_{\mathrm{N},\mathrm{m}}\end{bmatrix}_{\mathrm{N}\times\mathrm{m}} \tag{10}$$

Let X_i represent the ith pelican, and X represents the pelican population matrix.Each member of the population in the suggested POA is a pelican, which is a potential fix for the stated issue. Consequently, we can use each of the potential solutions to evaluate the objective function of the given problem. Eq. (11) uses a vector known as the objective function vector to derive the values obtained for the objective function.

$$\mathrm{F}=\begin{bmatrix}\mathrm{F}_1\\ \vdots\\ \mathrm{F}_\mathrm{i}\\ \vdots\\ \mathrm{F}_\mathrm{N}\end{bmatrix}_{\mathrm{N}\times1}=\begin{bmatrix}\mathrm{F}(\mathrm{X}_1)\\ \vdots\\ \mathrm{F}(\mathrm{X}_\mathrm{i})\\ \vdots\\ \mathrm{F}(\mathrm{X}_\mathrm{N})\end{bmatrix}_{\mathrm{N}\times1} \tag{11}$$

where F_i is the objective function value of the ith candidate solution, and F is the objective function vector, To update potential answers, the suggested POA mimics pelicans' hunting and attack tactics and behavior. There are two stages to simulating this hunting strategy:

1. Approaching the prey (exploration phase).
2. Winging is done during the exploitation phase on the water's surface.

Phase 1: The Discovery Phase: Approaching the Prey The initial stage of the process involves the pelicans locating the prey and then approaching it. Modeling this pelican's approach enables it to scan the search space and explore various search space regions. The key to POA lies in randomizing the prey's position within the search space. As a result, POA has more exploration capacity while precisely searching the problem-solving domain. Eq. (12) simulates the aforementioned ideas as well as the pelican's analytical approach to the location of prey.

$$\mathrm{x}_{\mathrm{i},\mathrm{j}}^{\mathrm{P}_1}=\begin{cases}\mathrm{x}_{\mathrm{i},\mathrm{j}}+\mathrm{rand}.\left(\mathrm{p}_\mathrm{j}-\mathrm{I}.\mathrm{x}_{\mathrm{i},\mathrm{j}}\right), & \mathrm{F}_\mathrm{p}<\mathrm{F}_\mathrm{i}\\ \mathrm{x}_{\mathrm{i},\mathrm{j}}+\mathrm{rand}.\left(\mathrm{x}_{\mathrm{i},\mathrm{j}}-\mathrm{p}_\mathrm{j}\right), & \mathrm{else}\end{cases} \tag{12}$$

where p_j is the position of prey in the jth dimension, F_p is its objective function value, and $\mathrm{x}_{\mathrm{i},\mathrm{j}}^{\mathrm{P}_1}$ is the new status of the ith pelican in the jth dimension based on phase 1. The parameter I is a number that can arbitrarily equal 1 or 2. We choose a random parameter for each member and iteration. A value of two for this parameter further displaces a member, potentially leading them into previously unexplored regions of the search space. As a result, parameter I influences the POA's ability to explore and precisely scan the search space. We accept the new location for a pelican if the suggested POA enhances the value of the objective function. This type of update, known as effective updating, keeps the algorithm from going into sub optimal regions. We model this process using an Eq. (13) [52].

$$\mathrm{X}_\mathrm{i}=\begin{cases}\mathrm{X}_\mathrm{i}^{\mathrm{P}_1}, \mathrm{F}_\mathrm{i}^{\mathrm{P}_1}<\mathrm{F}_\mathrm{i}\\ \mathrm{X}_\mathrm{i} \quad \mathrm{else}\end{cases} \tag{13}$$

where $\mathrm{F}_\mathrm{i}^{\mathrm{P}_1}$ is the pelican's objective function value based on phase 1, and $\mathrm{X}_\mathrm{i}^{\mathrm{P}_1}$ is its new status of the ith pelican.

Phase 2: The Exploitation Phase, or Winging on the Water Surface In the second stage, the pelicans gather their prey in their throat pouches after reaching the water's surface and spreading their wings to push the fish upward. This tactic helps pelicans catch more fish in the attacked region. As a result of simulating this pelican behavior, the suggested POA converges to more advantageous locations within the hunting region. This procedure

boosts POA's capacity for local search and exploitation. From a mathematical perspective, the algorithm must look at the points surrounding the pelican position in order to converge on an optimal solution. Eq. (14) mathematically simulates pelicans' hunting behavior.

$$x_{i,j}^{p_2}=x_{i,j}+R.\left(1-\frac{t}{T}\right) \tag{14}$$

where $R.\left(1-\frac{t}{T}\right)$ is the neighborhood radius of $x_{i,j}$, t is the iteration counter, and T is the maximum number of iterations. Additionally, $x_{i,j}^{p_2}$ is the updated status of the ith pelican in the jth dimension based on phase 2. The radius of the neighborhood of the population members to search locally near each member to converge to a better solution is represented by the coefficient $R.\left(1-\frac{t}{T}\right)$. This coefficient effectively enhances the POA exploitation power by bringing the population closer to the ideal global solution. Initially, a larger region surrounds each member due to the high value of this coefficient. The $R.\left(1-\frac{t}{T}\right)$ coefficient falls as the method replicates more, resulting in smaller radii for each member's neighborhood. This allows us to scan the region surrounding each member of the population in smaller and more precise steps, ensuring the POA converges to solutions that are closer to the global (and even precisely global) optimal based on the usage concept. Eq. (15) models the new pelican posture, which has also undergone effective updating and received acceptance or rejection.

$$X_i=\begin{cases} X_i^{p_2}, F_i^{p_2}<F_i \\ X_i \quad \text{else} \end{cases} \tag{15}$$

where $F_i^{p_2}$ is the pelican's objective function value based on phase 2, and $x_i^{p_2}$ is its new status.

5. The proposed improving of POA

The Meta-heuristic algorithms are initially proposed, it was used mostly for continuous optimization problems. But mixed integer programming problems are not always problems with continuous variables , which means they are discrete variables . Binary optimization problems are a special case of discrete optimization, in which the search space variable is binary values [57, 58].

The transfer function plays a role in the optimizer binarization process for mapping the search space, optimizing the convergence speed and accuracy of the optimization algorithm. Under the transfer function, the position of the vector value is altered from 0 instead of 1 and vice versa. In this step, the transfer function will give a force to impel the particle in the binary space [59, 60].

Taking into consideration the critical need to use transfer functions to discretize the binary optimization problem, it makes sense to pay attention to the problem of discussing and exploring the design of transfer functions with regard to their diversity and effectiveness in practice. This being the case, the currently existing V-shaped transfer functions and S-shaped transfer functions [61, 62, 63] are initially revisited in this section.

In this paper we applied the new type of the family of transfer functions proposed by [64] which was known as tent-shaped transfer functions. According to this thought, a binary pelican optimization algorithm (BPOA) is suggested to mixed integer programming problems.

The V-shaped and S-shaped functions are the most common and widely used in the transfer functions [63]. The most representative V-shaped functions include V_1 to V_4, and the most representative S-shaped functions include functions S1 to S4. The mathematical equations of these functions are presented in Table 1 and the curves of these functions are presented in Figure 1 and 2. They bring it out clear that S-shaped functions are a category of fundamental functions that are generated out of the exponential function.

The positions will be updated in the S-shaped functions using Eq. (16).

$$x_i^k(t+1)=\begin{cases} 1 & \text{if } r < S\text{ function } (x_i^k(t+1)) \\ 0 & \text{if } r \geq S\text{ function } (x_i^k(t+1)) \end{cases} \tag{16}$$

where r is a random number $\in[0,1]$.

The positions will be updated in the V-shaped functions using Eq. (17).

$$x_i^k(t+1) = \begin{cases} x_i^k(t)^{-1} & \text{if } r < V\text{ function } (x_i^k(t+1)) \\ x_i^k(t) & \text{if } r \geq V\text{ function } (x_i^k(t+1)) \end{cases} \tag{17}$$

Table 1. Families of transfer functions

S-Shaped family		V-Shaped family	
S_1	$S_1(x) = \frac{1}{1+e^{-2x}}$	V_1	$V_1(x) = \left\lvert \operatorname{erf}\left(\frac{\sqrt{\pi}}{2}x\right)\right\rvert$
S_2	$S_2(x) = \frac{1}{1+e^{-x}}$	V_2	$V_2(x) = \lvert\tanh(x)\rvert$
S_3	$S_3(x) = \frac{1}{1+e^{-\frac{x}{2}}}$	V_3	$V_3(x) = \left\lvert \frac{x}{\sqrt{1+x^2}}\right\rvert$
S_4	$S_4(x) = \frac{1}{1+e^{-\frac{x}{3}}}$	V_4	$V_4(x) = \left\lvert \frac{2}{\pi}\arctan\left(\frac{\pi}{2}x\right)\right\rvert$

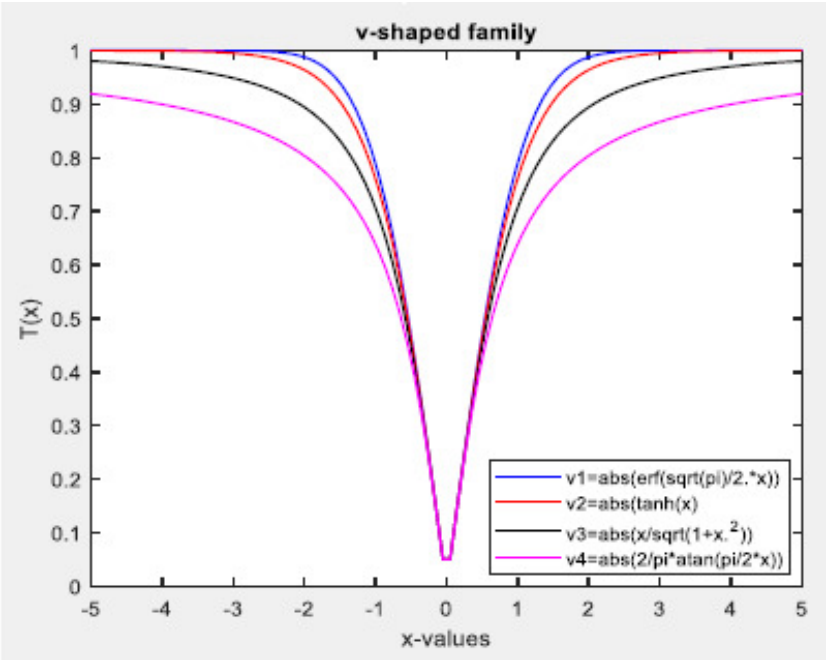

Figure 1. V-Shaped transfer function.

Today the existing transfer functions and their applicability [62, 65] are judged primarily by the results in binary optimization problems (BOPs). Resting on this fact, it becomes obvious that the attempts to develop a range of transfer functions can be quite helpful, not only does it offer more possibilities to discretize BOPs, but also it helps to gain more experience necessary to formulate a comprehensive theory that helps evaluate those functions. V-shaped transfer functions and S-shaped transfer functions [61] are composed of a trigonometric function, an anti-trigonometric function, an exponential function or a non-elementary function, thus, their computation needs more relative to a trigonometric function. This will affect the execution time of BOPs negatively. To this end, a new transfer function is suggested. Because the shape of the new transfer function is similar to the one of a tent, it can be referred to as a tent-shaped transfer function and is as follows [64]:

$$TT(x) = \frac{1}{(1+|x|)^R} \tag{18}$$

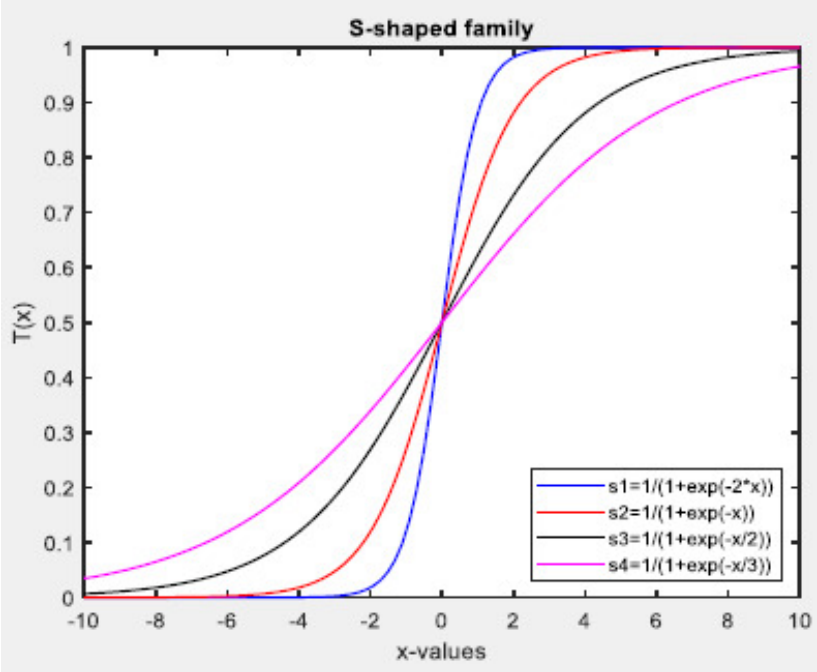

Figure 2. S-Shaped transfer function.

And R being a positive real number. With a suitable value of R a system of Tent functions family is obtained which have their expressions and Figures in Table 2 and Figure 3, respectively.

It is evident in Figures 2 and 3 that properties of the tent shaped functions are totally divergent to the S-shaped transfer functions. However, when we compare our function to the V-shaped transfer functions we find that they are only similar in that they both are symmetrical about the Y-axis, but they are quite different in a number of properties, including (but not limited to): The V-shaped transfer function is decreasing on $(-\infty, 0]$ increasing on$[0,\infty)$, and the reverse is true, it is obvious by looking at Figures 1 and 3 that the tent-shaped transfer functional is an inverted V-shaped transfer functional. This distinction might be insignificant compared with the following [64]:

1. All T-shaped transfer functions possess a unified mathematical formula. R represents a positive real number, and their computations are quite simple. Conversely, the V-shaped transfer functions lack a common calculation formula; furthermore, the mathematical formula for functions V_2 and V_4 are somewhat complicated.
2. The increase or decrease in the curvature of the T-shaped transfer functions is substantially more gradual than the curvature of the V-shaped transfer functions. This disparity in mathematical attributes is the most crucial element among the differing properties between T-shaped and V-shaped transfer functions.

Based on the two points above, the computational difficulty of T-shaped transfer functions is significantly less than that of V-shaped transfer functions.

6. Experimental Results and discussion

The improved mixed-integer pelican optimization algorithm (MI-POA) is applied in this section to eight integer and mixed-integer optimization problems. These problems are borrowed in various sources of the literature [66, 67, 68] and are shown in Table 3. And outline of these test problems is in the Appendix.

Table 2. T-Shaped transfer functions

Name	Transfer function
TT_1	$TT_1(x) = \frac{1}{(1+\lvert x\rvert)^{\frac{1}{2}}}$
TT_2	$TT_2(x) = \frac{1}{(1+\lvert x\rvert)}$
TT_3	$TT_3(x) = \frac{1}{(1+\lvert x\rvert)^2}$
TT_4	$TT_4(x) = \frac{1}{(1+\lvert x\rvert)^3}$

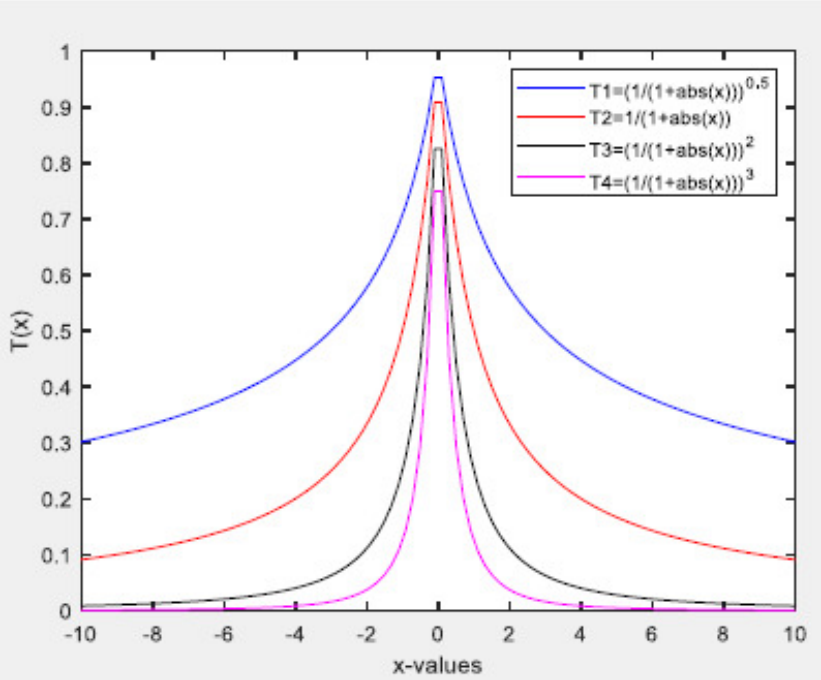

Figure 3. T-Shaped transfer function.

Table 3. Properties of the integer and mixed integer problems

Problems	Number of variables	Number of Integer variables	Number of constraints	Optimal
P_1	2	1	2	2
P_2	2	1	2	2.124
P_3	3	1	3	1.07654
P_4	4	4	1	-6
P_5	3	1	4	99.245209
P_6	7	4	9	3.557463
P_7	8	8	4	0.94347
P_8	5	3	4	7.667

In this paper, these problems have been addressed in the current paper using improved algorithm MI-POA. Overall, 100 runs of improved algorithm (MI-POA) of all the transfer functions considered in this paper have been performed on each problem. A successful run is defined as a successful run when the value of the error in the objective function value (error =|known objective function value-obtained objective function value) is less than 0.01. This limit is assumed as in [67, 68] . All the experiments were carried out on MATLAB 2023. The parameters of the POA algorithm have been defined with their respective values in Table 4.

Tables 5 and 6 measure the performance of the improved algorithm in terms of some statistical measures: best, worst, mean and standard deviation of objective function values achieved in runs.

Table 4. Parameters of the POA algorithm

Parameter	**Definition**	**Values**
N	Population size	30
T	Maximum number of iterations	500
I	Randomly selected per member and iteration	1 or 2
R	Constant	0.2
t	Current iteration number	1 to T

Table 5. Results obtained by MI-POA for P_1, P_2, P_3, P_4

Problems	Transfer function	Best	Worst	Mean	S.D
P_1	S_1	2	2.1644	2.0526	0.1228
	S_2	2	2.1673	2.0538	0.1294
	S_3	2	2.1755	2.0623	0.1317
	S_4	2	2.1691	2.0564	0.1309
	V_1	2	2.1382	2.0248	0.1043
	V_2	2	2.1361	2.0240	0.1040
	V_3	2	2.1358	2.0239	0.1033
	V_4	2	2.1332	2.0207	0.1011
	TT_1	2	2.0154	2.0066	0.0245
	TT_2	2	2.0151	2.0063	0.0219
	TT_3	2	2.0136	2.0048	0.0175
	TT_4	2	2.0147	2.0051	0.0182
P_2	S_1	2.124	2.1258	2.1246	0.0736
	S_2	2.124	2.1254	2.1240	0.0721
	S_3	2.124	2.1261	2.1253	0.0740
	S_4	2.124	2.1266	2.1259	0.0746
	V_1	2.124	2.1250	2.1248	0.0440
	V_2	2.124	2.12491	2.12431	0.04382
	V_3	2.124	2.12485	2.12423	0.03821
	V_4	2.124	2.12443	2.12411	0.03256
	TT_1	2.124	2.12429	2.12403	0.02501
	TT_2	2.124	2.12415	2.124005	0.02384
	TT_3	2.124	2.12422	2.124009	0.02478
	TT_4	2.124	2.12419	2.124003	0.02409
P_3	S_1	1.076543	1.07671	1.07665	0.05201
	S_2	1.0765427	1.0762	1.07659	0.05134
	S_3	1.076544	1.07663	1.07660	0.05182
	S_4	1.076542	1.07649	1.076448	0.05112
	V_1	1.076543	1.07645	1.076346	0.0365
	V_2	1.076543	1.076309	1.07648	0.0409
	V_3	1.076543	1.07640	1.07642	0.0332
	V_4	1.076543	1.076399	1.076441	0.0301
	TT_1	1.076543	1.07648	1.076537	0.0041
	TT_2	1.076543	1.076396	1.07652	0.0032
	TT_3	1.076543	1.07644	1.07653	0.0038
	TT_4	1.076543	1.07637	1.076504	0.0033
P_4	S_1	-6.00	-6.00	-6.00	0
	S_2	-6.00	-6.00	-6.00	0
	S_3	-6.00	-6.00	-6.00	0
	S_4	-6.00	-6.00	-6.00	0
	V_1	-6.00	-6.00	-6.00	0
	V_2	-6.00	-6.00	-6.00	0
	V_3	-6.00	-6.00	-6.00	0
	V_4	-6.00	-6.00	-6.00	0
	TT_1	-6.00	-6.00	-6.00	0
	TT_2	-6.00	-6.00	-6.00	0
	TT_3	-6.00	-6.00	-6.00	0
	TT_4	-6.00	-6.00	-6.00	0

Based on Tables 5 and 6, we observe that the P_1 all transfer functions have the same optimum value of 2, which proves that they find the optimum. S shaped transfer function (S_1 to S_4) have greater Means ($\sim$2.052-2.062) and

Table 6. Results obtained by MI-POA for P5, P6, P7, P8

Problems	Transfer function	Best	Worst	Mean	S.D
P_5	S_1	99.24532	98.5331	98.6421	1.5207
	S_2	99.24525	98.5692	98.8734	1.5001
	S_3	99.24526	98.4783	98.7223	1.5034
	S_4	99.24525	98.4532	98.7112	1.5219
	V_1	99.24527	98.5953	98.8635	1.4712
	V_2	99.24524	98.86402	98.6655	1.3436
	V_3	99.24526	98.6123	98.4521	1.3076
	V_4	99.24522	98.8804	98.9232	1.2635
	TT_1	99.24521	98.8941	99.0536	1.0176
	TT_2	99.245219	98.8743	99.0722	1.0162
	TT_3	99.245206	98.9345	99.1089	1.0148
	TT_4	99.245208	98.9052	99.0922	1.0156
P_6	S_1	3.5577	4.4257	4.2516	0.3306
	S_2	3.5578	4.5213	4.2342	0.3132
	S_3	3.5574	4.6375	4.3861	0.4222
	S_4	3.5574	4.4503	4.4331	0.4365
	V_1	3.5576	4.2366	3.6672	0.3225
	V_2	3.5576	4.2401	3.6744	0.3318
	V_3	3.5576	4.2397	3.6701	0.3274
	V_4	3.5574	4.2065	3.6542	0.3129
	TT_1	3.5575	3.9266	3.6325	0.2356
	TT_2	3.5575	4.1632	3.6477	0.3004
	TT_3	3.5575	4.1153	3.63002	0.2426
	TT_4	3.5575	4.1467	3.6294	0.2495
P_7	S_1	-0.94347	-0.92194	-0.93452	0.0267
	S_2	-0.94347	-0.91543	-0.92231	0.0343
	S_3	-0.94347	-0.90962	-0.93652	0.0285
	S_4	-0.94347	-0.92236	-0.93383	0.0225
	V_1	-0.94347	-0.92451	-0.93162	0.0174
	V_2	-0.94347	-0.93272	-0.93352	0.00952
	V_3	-0.94347	-0.92924	-0.93248	0.0166
	V_4	-0.94347	-0.93152	-0.93227	0.00981
	TT_1	-0.94347	-0.93576	-0.93562	0.00632
	TT_2	-0.94347	-0.93887	-0.93874	0.00357
	TT_3	-0.94347	-0.93795	-0.93836	0.00442
	TT_4	-0.94347	-0.93842	-0.93982	0.00318
P_8	S_1	7.66792	7.28422	7.45561	1.6427
	S_2	7.66785	7.3288	7.4472	1.5113
	S_3	7.66788	7.3433	7.4396	1.3371
	S_4	7.667911	7.2458	7.4175	1.6142
	V_1	7.66784	7.4428	7.6639	0.9767
	V_2	7.66789	7.4954	7.5822	1.0123
	V_3	7.66784	7.4593	7.6651	0.9422
	V_4	7.66784	7.4674	7.6659	0.9351
	TT_1	7.66784	7.5322	7.6662	0.4591
	TT_2	7.66781	7.5643	7.6659	0.2266
	TT_3	7.66783	7.5778	7.66578	0.4673
	TT_4	7.667809	7.5769	7.6669	0.3228

S.D ($\sim$0.123-0.132), which represents greater variability. V shaped transfer function are slightly better (Means = -2.021 -2.025, S.D = -0.101-0.104), and TT shaped transfer function are dominant with Means (= -2.0048 -2.0066) and lowest S.D (= -0.0175 -0.0245). TT transfer functions show better stability of P_1 with TT_3 showing the narrowest spread (Worst=2.0136, S.D=0.0175).

With respect to P_2 all transfer functions (S_1-S_4, V_1-V_4, TT_1-TT_4) give the precise Best value of 2.124, which is consistent optimum discovery. S shaped transfer function exhibits Means of 2.1240-2.1259 and greater S.D (0.0721-0.0746), which indicates moderate instability. V shaped transfer function narrows to Means 2.12411-2.1248 with S.D 0.0326-0.0440, whereas TT shaped transfer function performs better at Means 2.12400-2.12403

and minimum S.D (0.0238-0.0248), headed by TT_2, TT_4 (Worst=2.12415-2.12419). TT shaped transfer functions dominate P_2 with high precision and low variance, which is suitable in metaheuristic applications in mixed-integer.

Also, we observe that all transfer functions of P_3 hit the very Best value (1.076543 or minor variants such as 1.0765427) but S shaped transfer functions have higher Means ($\sim$1.07645-1.07665) and S.D ($\sim$0.051-0.052) which are variable. V shaped transfer functions are better (Means = -1.07635-1.07648, S.D=0.030-0.041) and TT shaped transfer functions are shining (Means = -1.07650-1.07654, S.D=0.003-0.004), TT_2 being the best (Worst=1.076396, S.D=0.0032).

In the case of the P_4, all transfer functions (S_1-S_4, V_1-V_4, TT_1-TT_4) have the same Best, Worst, Mean (-6.00), and S.D (0) values, indicating complete convergence to zero variability between runs. This homogeneity establishes of problem P_4 trivial nature, where algorithms trivially achieve the optimum without going wrong.

From Table 6 concerning the P_5, all transfer functions reach Best values close to 99.245 but S shaped transfer function exhibit Means close to 98.64-98.87 with higher S.D (close to 1.50-1.52) indicating variability. V shaped transfer function improve to Means =98.45-98.92, S.D =1.26-1.47, TT shaped transfer function dominate with Means = 99.05-99.11, lowest S.D (=1.01-1.02) dominated by TT3 (Worst=98.9345).

Also the P_6 we see that all transfer functions converge to Best values around 3.557, however, S shaped transfer function converge to higher Means (4.23-4.43) and S.D ($\sim$0.31-0.44), and hence worse convergence. V shaped transfer function is much better (Means =3.65-3.67, S.D =0.31-0.33), whereas TT shaped transfer function dominates with Means =3.63-3.65, and smaller S.D (=0.24-0.30), especially TT_1 (Worst=3.9266).

In the P_7, all the transfer functions give the same Best value of -0.94347, but S shaped transfer functions give Means of -0.922 to -0.936 with a larger S.D ($\sim$0.023-0.034), which is a measure of variability. V shaped transfer function narrows to Means -0.931 -0.933 (S.D 0.0095-0.017) whereas TT shaped transfer function performs best with Means -0.935 -0.939 and smallest S.D (0.003-0.006) with TT_2 and TT_4 leading the way (Worst near -0.938).

With regard to the P_8, All transfer functions are Best values in the range of 7.6678-7.6679, but S shaped transfer functions have the highest Means (7.4175-7.4556) and largest S.D ($\sim$1.34-1.64), which implies a high level of variability and worse convergence. V shaped transfer function improve to Means =7.58-7.666 (S.D=0.94-1.01), and TT shaped transfer function dominate with Means=7.665-7.667 and lowest S.D= ($\sim$0.23-0.47), and TT_2 led (Worst=7.5643, S.D=0.227).

Overall, TT shaped transfer function dominance has the tightest consistency, zero S.D on P_4, approximate 0.003-0.02 on problems (P_2-P_3/P_7), and minimum spreads of approximately 0.2-0.5 on P_5-P_6/P_8. V shaped transfer function second with moderate betterment than S shaped transfer function and S shaped transfer function exhibits most variability (S.D =0.05-1.6) which is best in exploration but not precision. Table 7 explains best transfer function , lowest S.D and highest S.D of every problems, and figure 4 explain standard deviation with each problem.

Table 7. best transfer function, lowest S.D and highest S.D of every problems

Problem	Best transfer function	Lowest S.D	Highest S.D
P1	TT_3 (0.0175)	TT (0.0175-0.0245)	S (0.1228-0.1317)
P2	TT_2/$_4$ ($\sim$0.024)	TT (0.0238-0.0250)	S (0.0721-0.0746)
P3	TT_2 (0.0032)	TT (0.0032-0.0041)	S (0.0511-0.0520)
P4	All (0)	All (0)	All (0)
P5	TT_3 (1.0148)	TT (1.0148-1.0176)	S (1.5001-1.5219)
P6	TT_1 (0.2356)	TT (0.2356-0.3004)	S (0.3132-0.4365)
P7	TT_2 (0.0036)	TT (0.0032-0.0063)	S (0.0225-0.0343)
P8	TT_2 (0.2266)	TT (0.2266-0.4673)	S (1.3371-1.6427)

Based on Figure 4, the TT-Shaped transfer function have the lowest standard deviation in all problems, which implies a high convergence irrespective of the initialization.

Figures 5-6 show the performance of S-Shaped transfer function , V-Shaped transfer function and TT-Shaped transfer function in terms of time that represents the average computation time in seconds used by the all algorithms in the case of successful runs. We can see how the TT-Shaped transfer function is better than the S-Shaped transfer function and V-Shaped transfer function.

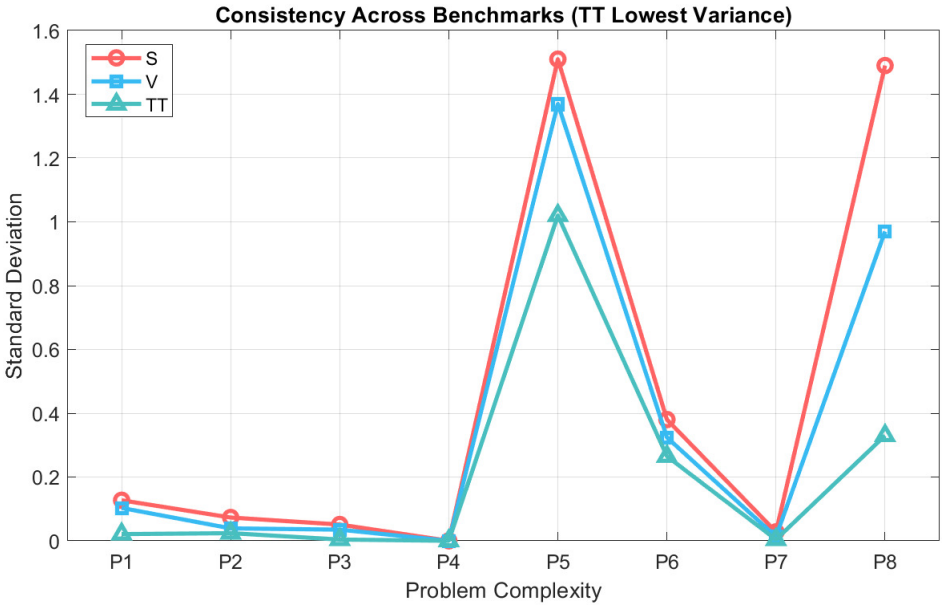

Figure 4. S.D values of each Problem.

In order to further confirm the performance of MI-POATT_4 with two evolutionary algorithms are compared: Black hole algorithm (BHA) and Flower pollination algorithm (FPA). Experimental results obtained by three algorithms are given in Table 8, where (PS) is success rate representing percentage of the successful runs to total runs for the algorithm, and (FE) represents average number of objective function evaluations used by the algorithm in successful runs. According to the results in Table 8, it is clear that MI-POATT_4 algorithm can obtain the optimal solution to different kinds of linear or nonlinear mixed integer programming problems with a success rate of 100 percent of all test functions except problem 6 with 92 %, which proves the stability and robustness of MI-POATT_4. in order to measure the efficiency of MI-POATT_4 in solving all test problems, the sum of the mean FE provided by BHA and FPA algorithms are 36439 and 28729 while the sum of the FE produced by MI-POA is 8321. In this way, MI-POA is considered to be the most effective and with the best minimum total (FE). Figure 7 and 8 represent the performance MI-POATT_4, BHA and FPA in terms of total function evaluation and total number of success of all test functions. Overall, MI-POATT_4 is better than any of the compared EAs algorithms regarding the measure of success rate and efficiency.

Table 8. Results obtained by three algorithms

Problems	MI-POA TT_4			BHA			FPA		
	FE	PS	t	FE	PS	t	FE	PS	t
P1	330	100	0.0402	256	76	1.3124	198	83	2.5412
P2	140	100	0.0697	130	74	1.3767	567	65	2.7311
P3	1572	100	0.2822	4487	69	3.6477	5437	77	2.8932
P4	133	100	0.0053	375	100	0.8479	120	100	1.2184
P5	2421	100	0.7448	12450	65	2.3674	6387	72	1.7752
P6	4630	92	1.4523	8576	55	5.7312	3746	67	7.3444
P7	800	100	1.7201	687	77	4.2287	7644	100	3.0276
P8	225	100	0.5541	488	100	3.3651	430	100	6.1067

A statistical analysis is conducted to further evaluate the performance of the TT4 method. The Wilcoxon signed-rank test is a non-parametric statistical test. We use the Wilcoxon signed-rank test (WST) to find the important difference between TT4 and the other methods that were talked about. The WST results are presented in Table 9, with a significance level of$\alpha = 0.05$. As shown in Table 9, the statistical significance of the proposed method is better that of other methods and the p-value is less than 0.05 for all methods.

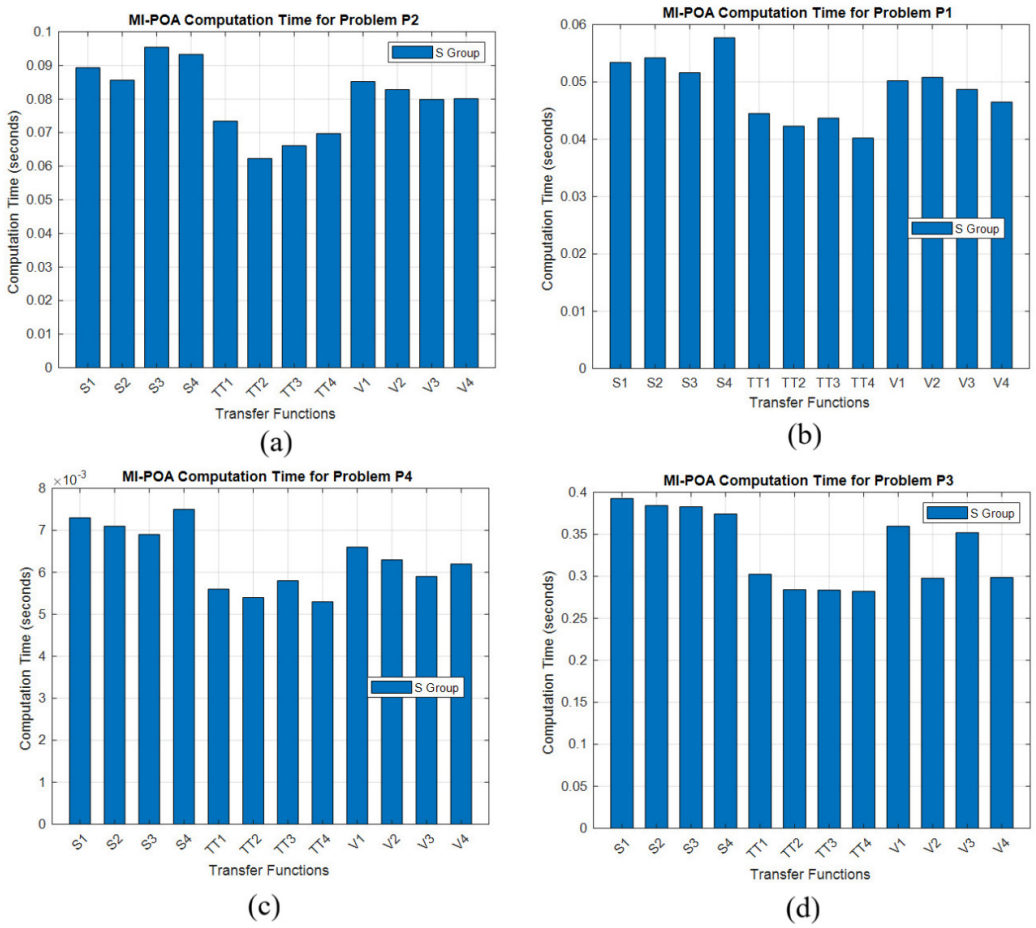

Figure 5. computational time (a) P1 , (b) P2 , (c) P3 , and (d) P4.

Table 9. p-values for the Wilcoxon signed-rank test of the proposed method results with three competitor methods

Pairwise comparison	p-value
TT4 vs S1	0.0025
TT4 vs S2	0.0021
TT4 vs S3	0.0034
TT4 vs S4	0.0027
TT4 vs V1	0.0025
TT4 vs V2	0.0028
TT4 vs V3	0.0014
TT4 vs V4	0.0027

7. Conclusion

This study successfully developed and validated MI-POA, an enhanced OPA tailored for integer and mixed-integer linear programming problems, by integrating novel tent-shaped transfer functions that outperform traditional S- and V-shaped alternatives in discretization efficiency and solution precision. Experimental results across eight benchmark problems confirm MI-POA's superior performance, with tent functions (TT1-TT4) achieving the lowest standard deviations such as 0.0032 for P3, means closest to optima, such as 99.1089 for P5, and perfect optimality on simpler cases, such as P4 at -6.00, demonstrating robust exploration-exploitation balance and computational

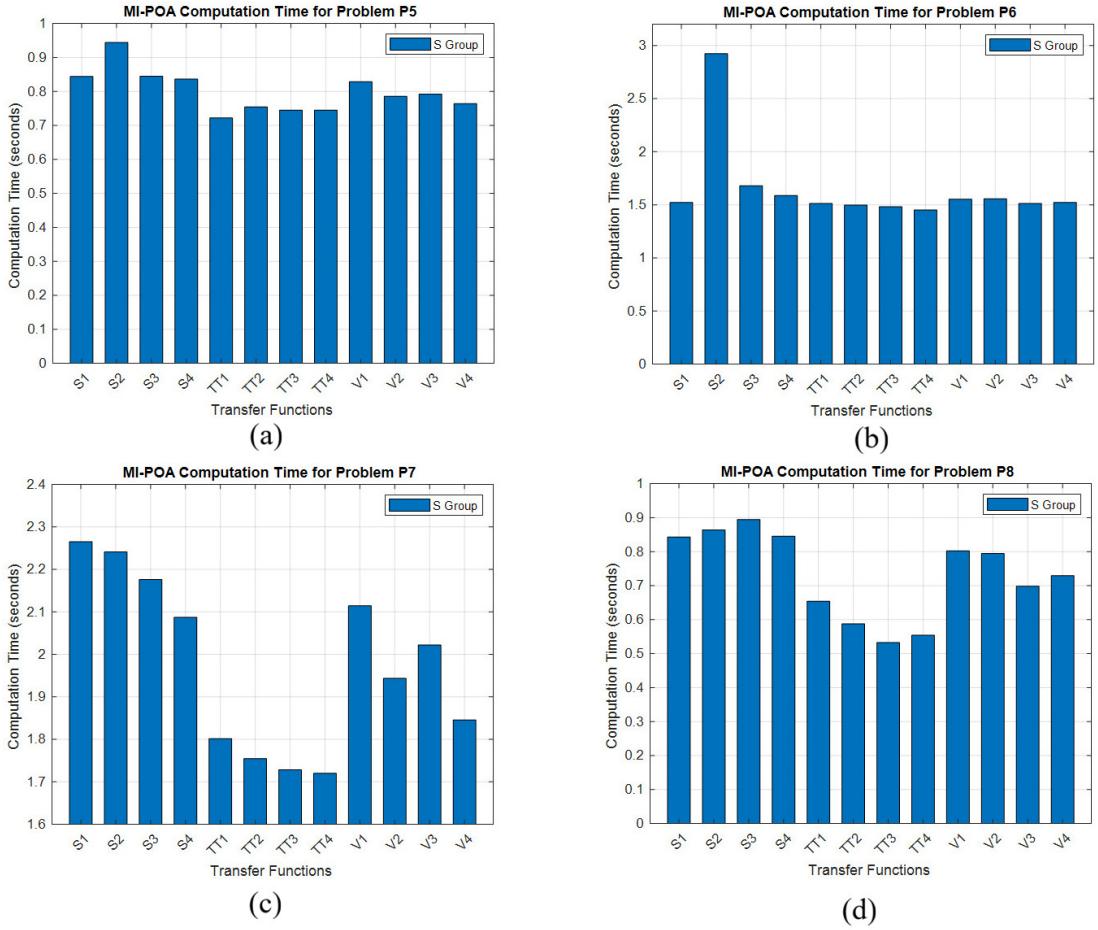

Figure 6. computational time of (a) P5 , (b) P6 , (c) P7 , and (d) P8.

simplicity. MI-POA advances metaheuristic solvers for NP-hard real-world applications in scheduling, logistics, and resource allocation, offering a promising, scalable alternative to exact methods like branch-and-bound. Future work will explore hybridizations and large-scale implementations.

Appendix

Problem (1)

$$\min f = 2x+y$$

subject to:

$$1.25-x^2 - y \leq 0$$

$$x+y \leq 1.6$$

$$0 \leq x \leq 1.6$$

$$y \in \{0, 1\}$$

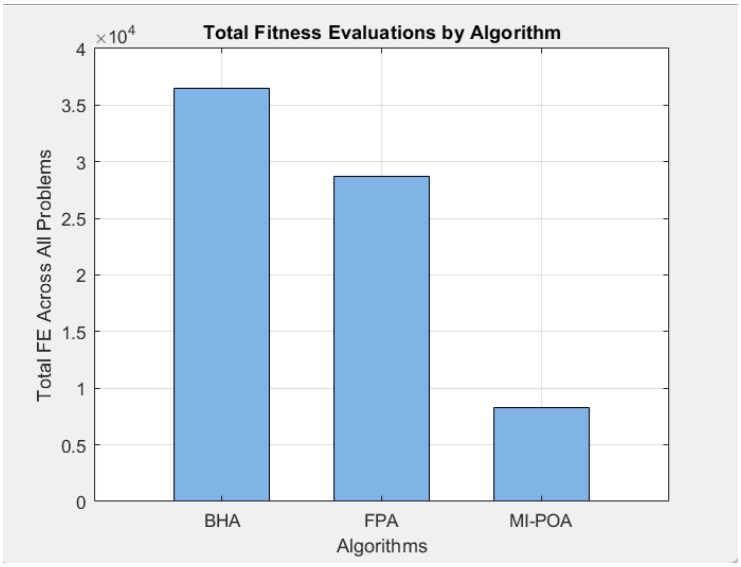

Figure 7. The performance MI-POA TT_4 , BHA and FPA in terms of total function evaluation

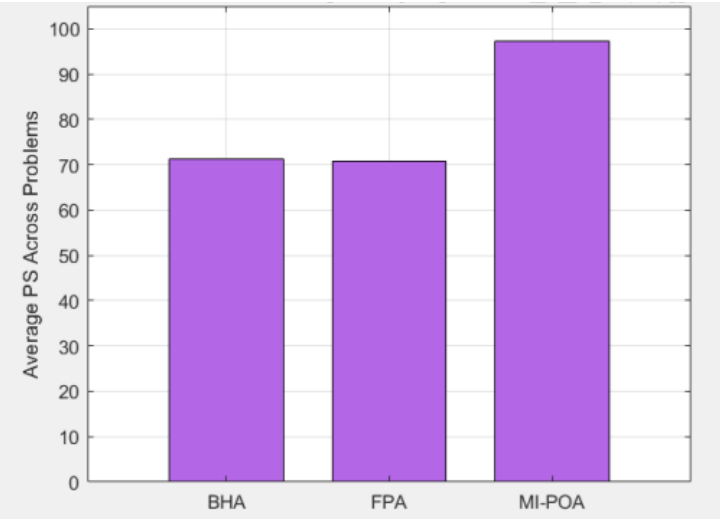

Figure 8. The performance MI-POA TT_4 , BHA and FPA in terms of total number of success

The optimal solution is:

$$\mathrm{f}= 2\ ,\ (\mathrm{x},\mathrm{y}) = (0.5,\ 1)$$

Problem (2)

$$\mathrm{minf}= -\mathrm{y}+2\mathrm{x}- \ln\left(\frac{\mathrm{x}}{2}\right)$$

subject to:

$$-\mathrm{x} - \ln\left(\frac{\mathrm{x}}{2}\right) +\mathrm{y}\leq 0$$

$$0.5 \leq x \leq 1.5$$
$$y \in \{0, 1\}$$

The optimal solution is:

$$f = 2.124\ ,\ (x, y) = (1.375,\ 1)$$

Problem (3)

$$\min f = -0.7y + 5(x_1 - 0.5)^2 + 0.8$$

subject to:

$$-\exp(x_1 - 0.2)\ - x_2 \leq 0$$
$$x_2 + 1.1y \leq -1$$
$$x_1 - 1.2y \leq 0.2$$
$$0.2 \leq x_1 \leq 1$$
$$-2.22554 \leq x_2 \leq -1$$
$$y \in \{0, 1\}$$

The optimal solution is:

$$f = 1.07654\ ,\ (x_1, x_2, y) = (0.94194,\ -2.1,\ 1)$$

Problem (4)

$$\min f = (x_1 + 2x_2 + 3x_3 - x_4)\,(2x_1 + 5x_2 + 3x_3 - 6x_4)$$

subject to:

$$x_1 + 2x_2 + x_3 + x_4 \leq 4$$
$$x_i \in \{0, 1\}\ ;\ i = 1, 2, 3, 4$$

The optimal solution is:

$$f = -6\ ,\ (x_1, x_2, x_3, x_4) = (0,\ 0,\ 1,\ 1)$$

Problem (5)

$$\min f = 7.5\ y + 5.5\,(1 - y) + 7v_1 + 6v_2 + 50\frac{y/(2y-1)}{0.9\,[1 - \exp(-0.5v_1)]} + 50\frac{1 - (y/(2y-1))}{0.8\,[1 - \exp(-0.4v_2)]}$$

subject to:

$$0.9\,[1 - \exp(-0.5v_1)] - 2y \leq 0$$
$$0.8\,[1 - \exp(-0.4v_1)] - 2(1 - y) \leq 0$$
$$v_1 \leq 10y$$
$$v_2 \leq 10(1 - y)$$
$$v_1,\ v_2 \geq 0$$
$$y \in \{0, 1\}$$

The optimal solution is:

$$f = 99.245209\ ,\ (y, v_1, v_2) = (1,\ 3.514237,\ 0)$$

Problem (6)

$$\text{minf}=(y_1-1)^2+(y_2-1)^2+(y_3-1)^2-\ln(y_4+1)+(x_1-1)^2+(x_2-2)^2+(x_3-3)^2$$

subject to:

$$y_1+y_2+y_3+x_1+x_2+x_3\leq 5$$

$$y_3^2+x_1^2+x_2^2+x_3^2\leq 5.5$$

$$y_1+x_1\leq 1.2$$

$$y_2+x_2\leq 1.8$$

$$y_3+x_3\leq 2.5$$

$$y_4+x_1\leq 1.2$$

$$y_2^2+x_2^2\leq 1.64$$

$$y_3^2+x_3^2\leq 4.25$$

$$y_2^2+x_3^2\leq 4.64$$

$$x_i\geq 0\ ;i=1,2,3$$

$$y_i\in\{0,1\}\ ;i=1,2,3,4$$

The optimal solution is:

$$f=3.557463\ ,\ (x_1,x_2,x_3,y_1,y_2,y_3,y_4)=(0.2,\ 1.280624,\ 1.954483,\ 1,\ 0,\ 0,\ 1)$$

Problem (7)

$$\text{minf}=r_1r_2r_3$$

$$\text{where}\ \ r_1=1-0.1^{y_1}0.2^{y_2}0.15^{y_3}$$

$$r_2=1-0.05^{y_4}0.2^{y_5}0.15^{y_6}$$

$$r_3=1-0.02^{y_7}0.06^{y_8}$$

subject to:

$$y_1+y_2+y_3\geq 1$$

$$y_4+y_5+y_6\geq 1$$

$$y_7+y_8\geq 1$$

$$3y_1+y_2+2y_3+3y_4+2y_5y_6+3y_7+2y_8\leq 10$$

$$y_i\in\{0,1\}\ ;i=1,2,3,4,5,6,7,8$$

The optimal solution is:

$$f=-0.94347\ ,\ (y_1,y_2,y_3,y_4,y_5,y_6,y_7,y_8)=(0,\ 1,\ 1,\ 1,\ 0,\ 1,\ 1,\ 0)$$

Problem (8)

$$\min f = 2x_1 + 3\,x_2 + 1.5y_1 + 2y_2 - 0.5y_3$$

subject to:

$$x_1^2 + y_1 = 1.25$$

$$x_2^{1.5} + 1.5y_2 = 3$$

$$x_1 + y_1 \leq 1.6$$

$$1.333x_2 + y_2 \leq 3$$

$$0 \leq x_i \leq 2; \, i = 1, 2$$

$$y_j \in \{0, 1\} \; ; j = 1, 2, 3$$

The optimal solution is:

$$f = 7.667 \; , \; (x_1, x_2, y_1, y_2, y_3) = (1.117, \; 1.310, \; 0, \; 1, \; 1)$$

REFERENCES

1. Wolsey, L. A. *Integer programming*, IIE Transactions, vol. 32, p. 273–285, 2000.
2. Khalil, A. W. *An improved flower pollination algorithm for solving integer programming problems*, International Journal of Applied Mathematics and Information Sciences, vol. 3, no. 1, p. 31-37, 2015.
3. Al-Fakih, A. M., Algamal, Z. Y., Lee, M. H., Aziz, M., & Ali, H. T. M. *QSAR classification model for diverse series of antifungal agents based on improved binary differential search algorithm*, SAR and QSAR in Environmental Research, vol.30, no.2, p.131-143, 2019.
4. Andu, Y., Lee, M. H., & Algamal, Z. Y. *Generalized dynamic principal component for monthly nonstationary stock market price in technology sector*, Journal of Physics: Conference Series, vol. 1132, p. 012076, 2018.
5. Ashford, R. *Mixed integer programming: A historical perspective with Xpress-MP*, Annals of Operations Research, vol. 149, no. 1, pp. 5–15, 2007.
6. Hussein, R., & Algamal, Z. Y. *Improving Set-union knapsack problem based on binary spotted hyena optimization algorithm*, Statistics, Optimization & Information Computing, vol. 15, no.1, p. 1653-1663, 2026.
7. Qasim, O. S., & Algamal, Z. Y. *A gray wolf algorithm for feature and parameter selection of support vector classification* , International Journal of Computing Science and Mathematics, vol. 13, 93-102, 2021.
8. Genova, K. *A heuristic algorithm for solving mixed integer problems*, Cybernetics and Information Technologies, vol. 11, no. 2, pp. 3–12, 2011.
9. Garey, M. R. & Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, New York, 1979.
10. Papadimitriou, C. H. & Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity*, Courier Corporation, 1998.
11. Bixby, R. & Rothberg, E. *Progress in computational mixed integer programming—A look back from the other side of the tipping point*, Annals of Operations Research, vol. 149, no. 1, pp. 37–41, 2007.
12. Silver, E. A. *An overview of heuristic solution methods*, Journal of the Operational Research Society, vol. 55, no. 9, pp. 936–956, 2004.
13. Balas, E., Schmieta, S. & Wallace, C. *Pivot and shift—A mixed integer programming heuristic*, Discrete Optimization, vol. 1, no. 1, pp. 3–12, 2004.
14. Danna, E., Rothberg, E. & Pape, C. L. *Exploring relaxation induced neighborhoods to improve MIP solutions*, Mathematical Programming, vol. 102, no. 1, pp. 71–90, 2005.
15. Fischetti, M. & Lodi, A. *Local branching*, Mathematical Programming, vol. 98, no. 1, pp. 23–47, 2003.
16. Gendreau, M. & Potvin, J.-Y. *Metaheuristics in combinatorial optimization*, Annals of Operations Research, vol. 140, no. 1, pp. 189–213, 2005.
17. Wilbaut, C. & Hanafi, S. *New convergent heuristics for 0–1 mixed integer programming*, European Journal of Operational Research, vol. 195, no. 1, pp. 62–74, 2009.
18. Vassilev, V. & Genova, K. *An algorithm of internal feasible directions for linear integer programming*, European Journal of Operational Research, vol. 52, no. 2, pp. 203–214, 1991.
19. Abdel-Raouf, O., Abdel-Baset, M. & El-Henawy, I. *A new hybrid flower pollination algorithm for solving constrained global optimization problems*, International Journal of Applied Operational Research, vol. 4, no. 2, pp. 1–13, 2014.
20. Abdel-Raouf, O., Abdel-Baset, M. & El-Henawy, I. *An improved chaotic bat algorithm for solving integer programming problems*, International Journal of Modern Education and Computer Science, vol. 6, no. 8, pp. 18–28, 2014.
21. Winston, W. L. *Operations Research: Applications and Algorithms*, Thomson Learning, Inc., 2004.

22. Gonzalez, M. *A hyper-matheuristic approach for solving mixed integer linear optimization models in the context of data envelopment analysis*, PeerJ Computer Science, vol. 8, p. e828, 2022.
23. Hochba, D. S. *Approximation algorithms for NP-hard problems*, ACM SIGACT News, vol. 28, no. 2, pp. 40–52, 1997.
24. Talbi, E.-G. *Combining metaheuristics with mathematical programming, constraint programming and machine learning*, Annals of Operations Research, vol. 240, no. 1, pp. 171–215, 2016.
25. Basheer, G., Mohammed, L., & Algamal, Z. Y. *Solving 0–1 knapsack problem by an improved binary monarch butterfly algorithm*, Statistics, Optimization & Information Computing, vol. 15, no.4, p. 2382-2396, 2026.
26. Basheer, G. T., & Algamal, Z. Y. *Improving flower pollination algorithm for solving 0–1 knapsack problem*, Journal of Physics: Conference Series, vol. 1879, no.2, p. 022097, 2021.
27. Sabbagh, M. S. & Soland, R. M. *An improved partial enumeration algorithm for integer programming problems*, Annals of Operations Research, vol. 166, pp. 147–161, 2009.
28. Land, A. H. & Doig, A. G. *An automatic method for solving discrete programming problems*, In *50 Years of Integer Programming 1958–2008*, Springer, pp. 105–132, 2009.
29. Gomory, R. E. *All-integer programming algorithm*, International Business Machines Corporation, 1960.
30. Wolsey, L. A. & Nemhauser, G. L. *Integer and Combinatorial Optimization*, John Wiley & Sons, 1999.
31. Chern, M.-S. & Jan, R.-H. *Reliability optimization problems with multiple constraints*, IEEE Transactions on Reliability, vol. 35, no. 4, pp. 431–436, 2007.
32. Lawler, E. L. & Bell, M. *A method for solving discrete optimization problems*, Operations Research, vol. 14, no. 6, pp. 1098–1112, 1966.
33. Gilmore, P. C. & Gomory, R. E. *The theory and computation of knapsack functions*, Operations Research, vol. 14, no. 6, pp. 1045–1074, 1966.
34. Balas, E. *An additive algorithm for solving linear programs with zero-one variables*, Operations Research, vol. 13, no. 4, pp. 517–546, 1965.
35. Sasaki, M., Kaburaki, S. & S. Yanagi, *System availability and optimum spare units*, IEEE Transactions on Reliability, vol. 26, no. 3, pp. 182–188, 2009.
36. Sabbagh, M. S. *A General Lexicographic Partial Enumeration Algorithm for the Solution of Integer Nonlinear Programming Problems*, Ph.D. Dissertation, 1985.
37. Srivastava, V. & Fahim, A. *A two-phase optimization procedure for integer programming problems*, Computers & Mathematics with Applications, vol. 42, no. 12, pp. 1585–1595, 2001.
38. Aardal, K., Weismantel, R. & Wolsey, L. A. *Non-standard approaches to integer programming*, Discrete Applied Mathematics, vol. 123, no. 1–3, pp. 5–74, 2002.
39. Ha C., & Kuo, W. *Reliability redundancy allocation: An improved realization for nonconvex nonlinear programming problems*, European Journal of Operational Research, vol. 171, no. 1, pp. 24–38, 2006.
40. Trojovský, P. & Dehghani, M. *Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications*, Sensors, vol. 22, no. 3, p. 855, 2022.
41. NC, K. *Pelican Optimization Algorithm for Optimal Demand Response in Islanded Active Distribution Network Considering Controllable Loads*, International Journal of Intelligent Engineering & Systems, vol. 15, no. 6, 2022.
42. Louchart, A., Tourment, N. & Carrier, J. *The earliest known pelican reveals 30 million years of evolutionary stasis in beak morphology*, Journal of Ornithology, vol. 152, no. 1, pp. 15–20, 2011.
43. Marchant, S. *Handbook of Australian, New Zealand & Antarctic Birds: Australian Pelican to Ducks*, Oxford University Press, 1990.
44. Mulkeen, S. & O'Connor, T. *Raptors in towns: towards an ecological model*, International Journal of Osteoarchaeology, vol. 7, no. 4, pp. 440–449, 1997.
45. Wilcoxon, F. *Individual comparisons by ranking methods*, Biometrics Bulletin, vol. 1, no. 6, pp. 80–83, 1945.
46. Lukman, A. F., Dawoud, I., Kibria, B. G., Algamal, Z. Y., & Aladeitan, B. *A new ridge-type estimator for the gamma regression model* Scientifica, vol. 1, p.5545356, 2021.
47. Algamal, Z., & Ali, H. M. *An efficient gene selection method for high-dimensional microarray data based on sparse logistic regression* , Electronic Journal of Applied Statistical Analysis, vol . 10 , no . 1 , 242-256, 2017.
48. Awwad, F. A., Odeniyi, K. A., Dawoud, I., Algamal, Z. Y., Abonazel, M. R., Kibria, B. G., & Eldin, E. T. *New two-parameter estimators for the logistic regression model with multicollinearity*, WSEAS Transactions on Mathematics, vol. 21, p.403-414, 2022
49. Al-Taweel, Y., & Algamal, Z. Y. *Some almost unbiased ridge regression estimators for the zero-inflated negative binomial regression model* Periodicals of Engineering and Natural Sciences, vol.8, no.1, p. 248-255, 2020.
50. Qasim, M. K., Algamal, Z. Y., & Ali, H. M. *A binary QSAR model for classifying neuraminidase inhibitors of influenza A viruses (H1N1) using the combined minimum redundancy maximum relevancy criterion with the sparse support vector machine*, SAR and QSAR in Environmental Research, vol. 29, no.(7), p.517-527, 2018
51. Anderson, J. G. *Foraging behavior of the American white pelican (Pelecanus erythrorhyncos) in western Nevada*, Colonial Waterbirds, pp. 166–172, 1991.
52. Zou, H. *Optimal scheduling of multi-energy complementary systems based on an improved pelican algorithm*, Energies, vol. 18, no. 2, p. 365, 2025.
53. AL-Taie, F. A. Y., Algamal, Z. Y., & Qasim, O. S. *A Hybrid Pelican Optimization Algorithm and Black Hole Algorithm for Kernel Semi-Parametric Fusion Modeling*, Fusion: Practice & Applications, vol. 11, no.1, p. 57-69, 2023.
54. AL-Taie, F. A. Y., Algamal, Z. Y., & Qasim, O. S. *Kernel semi-parametric model improvement based on quasi-oppositional learning pelican optimization algorithm*, Iraqi Journal for Computer Science and Mathematics, vol. 4, no.2, p. 156-164, 2023.
55. ZAYNAB, Z., Qasim, O. S., & Algamal, Z. Y. *Binary arithmetic optimization algorithm using a new transfer function for fusion modeling*, Fusion: Practice & Applications, vol. 18, no.2, p. 157-168, 2025.
56. Jamal, S. *Multi-objective optimal energy management of nanogrid using improved pelican optimization algorithm*, IEEE Access, vol. 12, pp. 41954–41966, 2024.

57. Bas, E. *BinDMO: a new Binary Dwarf Mongoose Optimization algorithm based on Z-shaped, U-shaped, and taper-shaped transfer functions for CEC-2017 benchmarks*, Neural Computing and Applications, vol. 36, no. 12, pp. 6903–6935, 2024.
58. Alkhateeb, A. N. & Al-Qazaz, Q. N. N. *Variable Selection in Weibull Accelerated Survival Model Based on Chaotic Sand Cat Swarm Algorithm*, Statistics, Optimization & Information Computing, vol. 13, no. 5, pp. 2105–2118, 2025.
59. Qasim, O. S. & Algamal, Z. Y. *Feature selection using different transfer functions for binary bat algorithm*, International Journal of Mathematical, Engineering and Management Sciences, vol. 5, no. 4, p. 697, 2020.
60. Kristiyanti, D. A., Sitanggang, I. S. & Nurdiati, S. *Feature selection using new version of V-shaped transfer function for salp swarm algorithm in sentiment analysis*, Computation, vol. 11, no. 3, p. 56, 2023.
61. Liu, J. *A binary differential search algorithm for the 0–1 multidimensional knapsack problem*, Applied Mathematical Modelling, vol. 40, no. 23–24, pp. 9788–9805, 2016.
62. Mafarja, M. *Binary dragonfly optimization for feature selection using time-varying transfer functions*, Knowledge-Based Systems, vol. 161, pp. 185–204, 2018.
63. Mafarja, M. *S-shaped vs. V-shaped transfer functions for ant lion optimization algorithm in feature selection problem*, Proceedings of the International Conference on Future Networks and Distributed Systems, 2017.
64. Almishlih, Z. A., Qasim, O. S. & Algamal, Z. Y. *Design and evaluation of a new tent-shaped transfer function using the Polar Lights Optimizer algorithm for feature selection*, Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska, vol. 15, no. 2, pp. 27–31, 2025.
65. Ghosh, K. K. *Binary social mimic optimization algorithm with x-shaped transfer function for feature selection*, IEEE Access, vol. 8, pp. 97890–97906, 2020.
66. Sun, Y. & Gao, Y. *An efficient modified particle swarm optimization algorithm for solving mixed-integer nonlinear programming problems*, International Journal of Computational Intelligence Systems, vol. 12, no. 2, pp. 530–543, 2019.
67. S. Gupta, S. & Deep, K. *An efficient grey wolf optimizer with opposition-based learning and chaotic local search for integer and mixed-integer optimization problems*, Arabian Journal for Science and Engineering, vol. 44, no. 8, pp. 7277–7296, 2019.
68. Deep, K. *A real coded genetic algorithm for solving integer and mixed integer optimization problems*, Applied Mathematics and Computation, vol. 212, no. 2, pp. 505–518, 2009.