A Metaheuristic Framework For The Pooling Problem: Application Of The Firefly Algorithm

Sana Akram[1], Huma Mehmood[1], Muhammad Farhan Tabassum[2, 3*], Sabah Iqbal[4], Anila Maqbool[1], Ayesha Qudus Saggu[5]

[1]Department of Mathematics, Lahore Garrison University, Lahore, 54000, PAKISTAN.
[2]Center for Skills Development & Leadership, University of Lahore, 54000, PAKISTAN.
[3]Department of Mathematics, University of Management and Technology, Lahore, 54000, PAKISTAN.
[4]Department of Mathematics, Forman Christian College University, Lahore, 54000, PAKISTAN.
[5]University Institute of Radiological Sciences and Medical Imaging Technology, Faculty of Allied Sciences, University of Lahore, Lahore, 54000, PAKISTAN.
[*]Corresponding Author: Muhammad Farhan Tabassum, farhanuet12@gmail.com

Abstract:
This paper investigates the optimization of pooling problems, especially the use of the Firefly Algorithm (FA), including a Proposed FA with self-adaptive properties, to solve the Haverly Pooling Problem in three distinct contexts. Using MATLAB simulations, the study evaluates the effectiveness of FA and Proposed FA in comparison to traditional optimization methods, including MSLP, MALT, and VNS. Pooling problems involve the combination of raw materials with various qualities to create final products that meet specific quality criteria, a task made more difficult by the non-linear complexity of the issue. Experiments on Haverly's pooling issues used the algorithms, and their results were contrasted with the exact answers. While the Proposed FA gets a near-optimal value of 400.25 for Haverly 1, making it almost undetectable from the exact solution, the exact answer is 400. Haverly 2's exact answer is 600; the Proposed FA, which shows a small overestimation of 0.87%, produces 605.23. With the exact answer of 750, Haverly 3 shows strong performance with the Proposed FA, which produces 748.96, only 0.14% lower than the accurate solution. The findings show that in every case the Proposed FA either exceeds or closely matches the exact solution, outperforming rival algorithms like MSLP, MALT, and VNS, which showed more variation. The Proposed FA's use of a self-adaptive step size improves the exploitation and exploration of the search space, hence producing very precise outcomes. This study finds that the Proposed FA is an effective optimization tool for solving pooling issues, showing improved performance compared to traditional optimization methods.

Keywords: pooling problem, firefly algorithm, Haverly's pooling problems, error analysis.

1. Introduction

Numerically, the optimization problems are in general form can be composed as:

$$\max_{x \epsilon \Re^d} f_i(x), \quad (i = 1,2,3 \dots, M), \tag{1}$$

$$\text{Subject to } h_j(x) = 0\,, \quad (j = 1,2,3 \dots, j), \tag{2}$$

$$g_k(x) \leq 0\,, \quad (k = 1,2,3 \dots, k), \tag{3}$$

Where $f_i(x), h_j(x)$ and $g_k(x)$ are elements of design vector $x = (x_1, x_2, x_3, \dots, x_d)^T$. Here the components x_i of x are known as they may be real, continuous, discrete and variable of design or decision or the

combination of both [1-5]. Functions $f_i(x)$ here $i = 1,2,3, \dots m$ are known as objective function otherwise essentially cost function, there is only one objective in case $M = 1$. The decision variable's coverage area is referred to as the $\Re^d$ design space or research space, created with the objective function's value which is referred to as the space of answer or solution space. The constraints are the equalities for h_j and inequalities for g_k [6-7]. Essentially in numbers of constraints $J + K$, we can also categorize optimization. There is no constraint at all in any case, $J = K = 0$, it is recognized that at that point as an unconstrained optimization problem. If $J = 0$, $K \geq 1$, become an inequality-constrain problem. where as $K = 0$, $J \geq 1$ it's called an equality-constrain problem. It's worth mentioning that equalities aren't clearly specified in certain formulations of the literature optimization, and only inequalities are supplied. Because of this, it is possible to define equality as two inequalities. $h(x) = 0$, for example, is identical to $h(x) \leq 0$ and $h(x) \geq 0$. Equality limitations, on the other hand, have unique characteristics that need specific treatment. One disadvantage which is the quantity of meeting in the search space, equality is virtually zero, making it extremely hard to get sample point to perfectly fulfill the equality. In reality, any type of authorization or allocation is employed [8-10].

We may also utilize the actual function processes for categorization [11-15]. Linear or nonlinear objective functions are possible. It becomes a linear issue when all of the restrictions h_j and $g_k$are linear. If all of the constraints and goal functions are linear, the issue becomes a linear programming issue. The term "programming" here does not refer to computer programming; rather, it refers to planning and optimization. In general, if all f_i, h_j, and $g_k$are nonlinear, we're dealing with a nonlinear optimization issue [1-5].

Introduced by Xin-She Yang in 2008, the Firefly Algorithm (FA) is a nature-inspired optimization tool. Inspired by the flashing behavior of fireflies, which exhibit unique light emission patterns for communication and mate attraction, it is a metaheuristic optimization algorithm. Every firefly's brightness indicates the quality of the solution for a particular optimization problem. Aiming to find the optimal solution inside the search space [16], the fireflies are attracted to one another depending on the degree of their luminosity.

Especially good at handling complex, high-dimensional, non-linear optimization problems is the FA; such problems occur in pooling scenarios where traditional optimization techniques could find it difficult to locate the global optimum. Driven by the intensity of their brightness, the technique works by gradually changing the positions of fireflies within the search space. The proximity between each firefly and its relative luminosity determines its appeal, which then suggests the quality of the solution it represents. Usually when the algorithm converges to a sufficiently optimal solution, this iterative process continues until a defined stopping criterion is met. Some recent practical applications of metaheuristics are reported in [17-29].

The Firefly Algorithm's main characteristic is its simplicity and flexibility. Unlike gradient-based methods, which often struggle with non-convex or multi-modal objective functions, the FA does not require derivatives of the objective function and can efficiently handle challenges with several local optima. This makes it a useful tool for tackling complex, real-world problems where finding the global optimum is challenging. The FA has been successfully applied in several fields including engineering design, machine learning, image processing, and financial modeling [16, 30].

The Firefly Algorithm's (FA) basic processes are the generation of an initial population of fireflies at random locations, evaluation of their fitness in line with the goal function, and progressive position adjustment based on light intensity and the relative locations of other fireflies. Two main components of the updating method are attraction—where fireflies draw towards brighter counterparts—and randomization, which promotes diversity and prevents early convergence. Driven by the brightness of their luminosity, the fireflies slowly gather around the optimal solution.

The ability of the Firefly Algorithm to handle large-scale optimization issues, non-linear, non-differentiable, and non-convex nature of the objective functions makes it an interesting option for handling pooling problems, where traditional approaches could struggle. This study shows how the FA can produce globally optimal solutions and looks at its application to pooling issues, particularly those connected to the mixing of raw materials with varying quality.

Pooling problems are a kind of optimization difficulty seen in industries where raw materials or ingredients with varied qualities must be combined to create a finished product meeting certain quality standard. These

problems are prevalent in food processing operations, chemical manufacture, and petroleum refining. The challenge is in the non-linear dynamics of combining many sources or inputs with different characteristics to meet production and quality criteria. In petroleum refining, crude oils with varied sulfur levels are mixed in a reservoir and the resulting combination has to meet the sulfur content criteria for various end products, including gasoline and diesel.

A classic example of a pooling problem is the one described by Haverly (1978), which was among the first to acknowledge the complex, non-linear nature of such problems. Haverly's conundrum combines many crude oil streams to create two refined products, each with specified sulfur content requirements [31]. The goal of the optimization task is to determine the optimal amounts of each crude oil to mix in order to maximize total profit, thereby following restrictions on sulfur content in the products and the availability of raw materials. Though relatively simple, this formulation revealed the complexity natural in pooling issues including the presence of several local optima and the difficulties of finding a global optimum using traditional methods [32].

Attributed to bilinear components in the governing equations, the non-convex character of these issues suggests that conventional optimization techniques including linear programming would not effectively find the global optimum. As a result, there is growing interest in using global optimization methods that can more effectively address the challenges generated by pooling. Among the recent approaches, the Firefly Algorithm (FA) has shown significant promise. Derived on the flashing behavior of fireflies, the FA is a metaheuristic optimization technique whereby the flash intensity affects the firefly's appeal, guiding it toward better solutions. Including pooling, this strategy has been successfully used to several optimization problems, including non-linear, multi-modal, and combinatorial ones.

This paper looks at how the Firefly Algorithm may be used on pooling issues, especially with regard to Haverly's case studies requiring optimal mixing techniques under several environments. The goal is to show how well the Firefly Algorithm overcomes the limitations of traditional approaches and achieves globally optimal solutions. We will outline the formulations of Haverly's pooling problems, investigate their nonlinear characteristics, and look at how the Firefly Algorithm can be adapted and applied to these problems to find solutions that traditional approaches might overlook.

2. Firefly Algorithm

Soft computing includes several computer techniques, including neural networks, fuzzy logic, evolutionary algorithms, chaos theory, among others. One of the most promising fields of artificial intelligence, Swarm Intelligence (SI), has seen notable relevance and popularity in recent years. The study of the behaviors of insects, worms, bees, fireflies, and aggregations of birds and fish inspires further SI field research. The coordinated activities of these companies show that by working together they can achieve their preferred goal [33-36].

Inspired by firefly behavior, Xin-She Yang created a new meta-heuristic algorithm called the Firefly Algorithm (FA) [37]. It relates totally to natural stochastic processes. It can handle issues like NP-Hard optimization. After looking over the collection for possible answers, this approach uses some randomization. Shows the fundamental structure of the firefly algorithm. Aiming to find the answer, the trial-and-error approach reflects a heuristic character. In certain cases, this does not ensure the best response [38].

2.1 Standard Firefly Algorithm

To create firefly-inspired algorithms, we can now idealize few of the strobes features of fireflies. We now use the suggestions below idealized guidelines for simplicity in defining the regular FA:

Pseudo Code: Firefly Algorithm
Objective function $F(\mathbf{x})$,$\mathbf{x} = (x_1, \dots, x_d)^T$.
Create primary n fireflies' population $\mathbf{x_i}(i = 1,2, \dots, n)$.
Light intensity I_i at $\mathbf{x_i}$is resolute by $F(\mathbf{x_i})$.
Define light absorption coefficient γ.
while$(t <$ MaxGeneration),
for $i = 1: n$ (all n fireflies)

for j = 1: n (all n fireflies) (inner loop)
if ($I_i < I_j$)
Move firefly **for** i towards j.
end if
Vary attractiveness with distance r via exp $[-\gamma r^2]$.
New solutions should be evaluated, and light intensity should be updated.
end for j
end for i
Rank the fireflies to determine which are the greatest in the world right now.g_*.
end while
Results of the post-production and visualization.

1. Because all fireflies are unisex, one firefly will be attracted to another firefly regardless of gender.
2. The attractiveness of anything is related to how brilliant it is of a firefly. Therefore, the one that is less brilliant would gravitate towards the one that is brighter for any two flashing fireflies. The attraction is related to brightness, both of which diminish as the distance between them grows. It will travel at random if no one is brighter than a single firefly.
3. The landscape of the goal function influences or determines the brightness of a firefly [39-44].

For example, the brightness might simply be proportional to the goal function's value an optimal solution. Other functions, comparable to the fitness function in genetic algorithms, types of brightness can be described. The FA's fundamental stages may be summed up in the pseudo code given above based on these three rules [45-48].

2.2 Proposed Firefly Algorithm

Renowned for their research on the human decision-making process, Boyd and Richerson typically integrate [36] this process into their work. One's own experiences and those of their neighbors. The other option is the present state. Inspired by this idea, we use it to control the motion of every firefly, so guiding its investigation inside the search space.

Self-Adaptability Property: Distinct difficulties, fireflies, and environmental elements should guide stage sets. Based on these ideas, a method is suggested that creates a framework dependent on every firefly's particular experience and present situation. Keeping a small size is the best strategy for the Firefly stage. In the same vein, the firefly stage has to be raised, much more distant from the ideal answer. Fireflies are used to balance the global search and local search between the above two. One should consider the historical relevance of the firefly stage as well as its present state. This thesis offers historical firefly data, including ideal values from its two most current versions. The step α of each firefly is defined, depending on the explanation and other studies, using Eq. (4) and Eq. (5), respectively [49-53]. Consider as:

$$h_i(t) = \frac{1}{\sqrt{(f_{pi}(t-1)-f_{pi}(t-2))^2+1}}, \quad (4)$$

$$\alpha_i(t+1) = \frac{1}{\sqrt{(f_{best}(t)-f_i(t))^2+h_i(t)^2+1}}, \quad (5)$$

Where $h_i(t)$ is the historical detail of ith firefly for past two iterations. The worth of the best in terms of fitness ith firefly solution is f_{pi}. $f_{best}$is the f_i is the fitness value of ith, which represents the current knowledge, and f_i is the fitness value of the best solution of population here to be discovered. Each firefly's the next iteration step is self-adaptive, defined by the distinction between its most recent fitness value and the best fitness value of the population. So, each firefly's stride might change with each iteration, and at the same iteration, the Each firefly's step is also recorded altered.

2.3 Firefly algorithm with self-adaptive property

Initialization: The firefly population is initialized, and their fitness is evaluated. The best fitness value (f_{best}) is also stored, representing the global optimum.

Iterative Process:

- For each firefly, historical information $h_i(t)$ is calculated based on the difference in the fitness values of the last two iterations.
- The self-adaptive step size $\alpha_i(t+1)$ is then computed using the current fitness value of the firefly, the best fitness value in the population, and the historical data.
- Fireflies are compared based on their fitness, and the firefly moves towards brighter (better) fireflies with a step size adjusted by $\alpha_i(t+1)$.

Position Update: The position of each firefly is updated by moving it towards the brighter firefly, with a small random perturbation to maintain diversity.

Global Best Update: The best solution is updated if a firefly has a better fitness than the current best solution.

Stopping Condition: The process continues until the maximum number of iterations is reached or another stopping condition is met.

Pseudo Code: Firefly algorithm with self-adaptive property

Initialize the population of fireflies
Evaluate the fitness of each firefly (f_i)
Set the best fitness value (f_{best}) in the population
For each iteration t = 1 to $max_{iterations}$:
For each firefly i in the population, Calculate the historical information $h_i(t)$

$$h_i(t) = \frac{1}{\sqrt{(f_{pi}(t-1) - f_{pi}(t-2))^2 + 1}}$$

Calculate the self-adaptive step size $\alpha_i(t+1)$

$$\alpha_i(t+1) = \frac{1}{\sqrt{(f_{best}(t) - f_i(t))^2 + h_i(t)^2 + 1}}$$

For each other firefly j in the population
If (f_i) < (f_j) (i.e., firefly i is brighter)
Move firefly i towards firefly j
Update the position of firefly i

$$P_i(t+1) = P_i(t) + \alpha_i(t+1) \cdot \left(P_j(t) - P_i(t)\right) + R_e$$

(where R_e is random effect introduces a perturbation to avoid local minima and improve the diversity of the population)
Evaluate the fitness of firefly i (f_i)
Update the global best solution
If (f_i) < (f_{best})
$f_{best} = f_i$
Store the position of firefly i as the best solution
If stopping condition met ($max_{iterations}$ or convergence)
Break
Return the global best solution (f_{best}) and its corresponding position P_{best}

The algorithm updates the step size dynamically based on each firefly's previous fitness values and the current best fitness of the population. This self-adaptive process helps balance exploration and exploitation by adjusting the search radius of each firefly depending on its current performance.

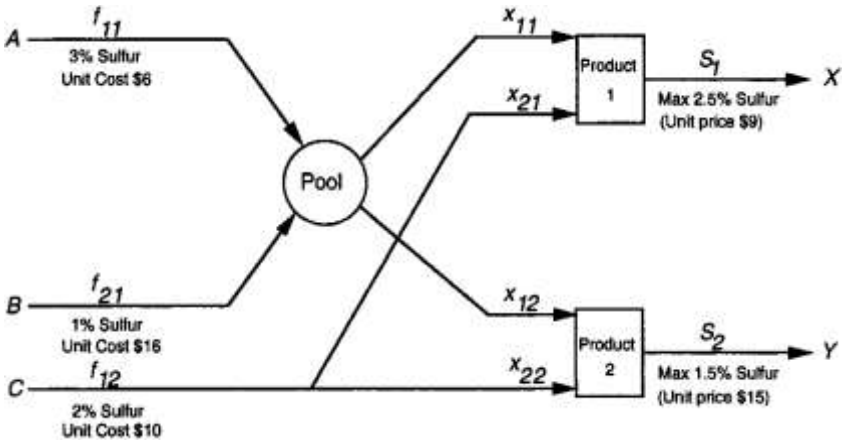
3. Pooling Problems

Pooling is a scheduling issue that happens when ingredients needed to make products are mixing together. Such as the process of mixing crude and refined petroleum Pooling takes place if sources, frequently, in a storage tank, are mixed together and the resultant mixture is distributed to a number of sites. In the synthesis of end products with diverse component quality specifications, the gathering and mixing of raw materials and stored goods is a crucial stage [54].

3.1 Haverly's pooling problem

In the famous small example by Haverly, pooling problem's nonlinear nature was first identified (1978). Two stages of the iterative algorithm were presented to demonstrate the potential difficulty of the problem. It entails calculating and repairing the damage intermediate pools' quality characteristics, then solving resulting linear program, it entails estimating and fixing the Stop; otherwise, it entails updating the values and repeating the procedure measures if the resultant attributes are the same as those predicted According to Haverly (1978), this approach does not lead to a global optimum [55-57]. Haverly has a slight pooling problem [58] are shown in Figure 1. Products are delivered to a single pool.

Figure 1 Pooling problem by Haverly



From two separate crude's oils streams, A and B. They differ in terms of sulfur content because A and B are different sources. A third supply C is blended directly with the two pool outputs rather than being fed into the pool. 3 percent for A, 1 percent for B, and 2 percent for C are the efficiency criteria for the brooks that flow into the pool. Products X and Y, which must be combined conform to Sulphur content requirements of 2.5 percent and 1.5 percent in that order, are formed by the mixing of fluxes from the supply stream C and the pool. The customer is determined by these constraints on the end-products X and Y. For instance, limitations may be the maximum sulfur content in gasoline in sense of the petroleum industry. $S_1 = 100$ and $S_2 = 200$, respectively, are the Maximum demand for goods X and Y, limiting the quantity of end-products generated [59-60].

The amounts of supply of A, B, and C, represented by the letters A, B, and C, are the variables in the preceding example f_{11}, f_{21}, and f_{12},as well as the final consistency the magnitudes of fluxes from pool products, represented by q, and the amount of sulfur in the pool as a consequence of mixing A and B, denoted by qx_{11} and x_{12}, in that order, and quantities of supply C, denoted by x_{21} and x_{22}, respectively, for products X and Y. From the values of the above variables, the It is simple to retrieve amounts of and end-products, as well as their final properties. On basis of assumption of a linear mixing model, it is then possible to formulate the problem in Figure 1 as

$$Minimize\ f = 6f_{11} + 16f_{21} + 10f_{12} - 9(x_{11} + x_{21}) - 15(x_{12} + x_{22}) \quad (6)$$

Subject to

$$f_{11} + f_{21} - x_{11} - x_{12} = 0 \quad (7)$$

$$f_{12} - x_{21} + x_{22} = 0 \quad (8)$$

$$q(x_{11} + x_{12}) - 3f_{11} - f_{21} = 0 \quad (9)$$

$$qx_{11} + 2x_{21} - 2.5(x_{11} + x_{21}) \leq 0 \quad (10)$$

$$qx_{12} + 2x_{22} - 1.5(x_{12} + x_{22}) \leq 0 \quad (11)$$

$$x_{11} + x_{21} \leq S_1 \quad (12)$$

$$x_{12} + x_{22} \leq S_2 \quad (13)$$

The Eq. (6) reflects the gap between the cost of input streams and the profit margins on items sold. Mass balances are expressed by equations (7) and (8). The pool quality sq, is expressed by Equation (9) in terms and of the input streams their characteristics. The consistency constraints on products are expressed by the equations (10) and (11). Finally, (12) and (13) ensure that no demands are met by the flows.
Constraints with bilinear terms (7-11) describe the issue of non-convexities most important to several local optima being available. Problem (6-13), for instance, has an objective function of 0 and has infinite local solutions, a local minimum of -100 with an objective function, and a single global optimum of -400 with an objective function. Typical nonlinear-programming method can therefore a large amount of sub-optimal solution, and for such problems. We need to investigate global optimization methods [60-64].

Problem 1: Haverly's pooling problem - Case I

Problem Formulation

$$Maximize\ \ Profit\ =\ 9x + 15y\ -\ 6A\ -\ 16B\ -\ 10(C_x\ +\ C_y)$$

Subject to

$$P_x + P_y - A - B\ =\ 0$$

$$x - P_x - C_x\ =\ 0$$

$$y\ -\ P_y - C_y\ =\ 0$$

$$p.P_x\ +\ 2C_x -\ 2.5x\ \leq 0$$

$$p.P_y\ +\ 2C_y -\ 1.5x\ \leq 0$$

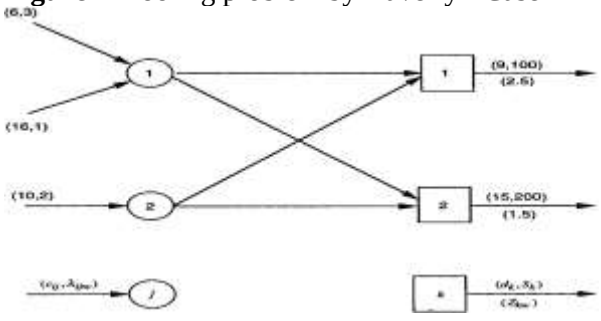
$$p.P_x\ +\ p.P_y -\ 3A -\ B\ =\ 0$$

$$x\ \leq\ 100$$

$$y\ \leq\ 200$$

where $p = 1, B = P_y = C_y = 100\ , y\ =\ 200$

Figure 2 Pooling problem by Haverly - Case I



Test Problem 2: Haverly's Pooling Problem - Case II
Problem formulation

The formulation for this problem is the same as for Test Problem 1 except that the upper bound on the product x is changed. The complete formulation is as follows:

$Maximize\ \ Profit\ =\ 9x+\ 15y\ -\ 6A\ -\ 16B\ -\ 10(C_x\ +\ C_y)$

Subject to

$P_x + P_y - A - B\ =\ 0$

$x - P_x - C_x\ =\ 0$

$y\ -\ P_y - C_y\ =\ 0$

$p.P_x\ +\ 2C_x -\ 2.5x\ \leq 0$

$p.P_y\ +\ 2C_y -\ 1.5x\ \leq 0$

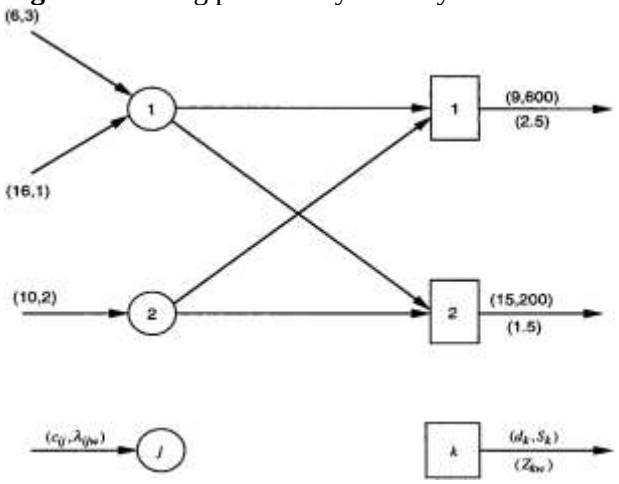
$p.P_x\ +\ p.P_y -\ 3A -\ B\ =\ 0$

$x\ \leq\ 600$

$y\ \leq\ 200$

where $p = 3, A = P_x = C_x = 300\ , y\ =\ 600$

Figure 3 Pooling problem by Haverly - Case II



Test Problem 3: Haverly's Pooling Problem - Case III

Problem formulation

The formulation for this problem is the same as for Test Problem 1 except that in the objective function the cost of B is changed from 16 to 13. The new formulation is:

$Maximize\ \ Profit\ =\ 9x+\ 15y\ -\ 6A\ -\ 13B\ -\ 10(C_x\ +\ C_y)$

Subject to

$P_x + P_y - A - B\ =\ 0$

$x - P_x - C_x\ =\ 0$

$y\ -\ P_y - C_y\ =\ 0$

$p.P_x\ +\ 2C_x -\ 2.5x\ \leq 0$

$p.P_y\ +\ 2C_y -\ 1.5x\ \leq 0$

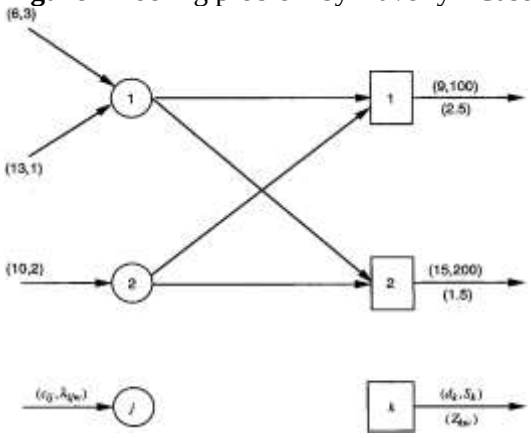
$p.P_x\ +\ p.P_y -\ 3A -\ B\ =\ 0$

$x\ \leq\ 100$

$y\ \leq\ 200$

where $p = 1.5,\ A = 50,\ B = 150,\ P_y = 200\ ,\ y\ =\ 200$

Figure 4 Pooling problem by Haverly - Case III



4. Parameter settings

The Firefly Algorithm (FA), an optimization technique inspired by nature, addresses optimization problems by emulating the behavior of fireflies. In pooling problems, the following parameters are crucial for amalgamating raw resources of varying grades to meet specific quality standards, hence ensuring the algorithm's efficient execution. The population size (N) denotes the quantity of fireflies within the population. A smaller population may lead to faster convergence but risks becoming trapped in local optima, whereas a larger population provides a more diverse search field. Typically, contingent upon the complexity of the issue, the population size may range from 50 to 200. The maximum iterations denote the frequency with which the algorithm executes, hence providing a termination criterion for the optimization procedure. The maximum iterations often range from 1000 to 5000, contingent upon the issue's complexity and the necessary precision.

Distance influences the fading of firefly light intensity via the light absorption coefficient γ. This setup controls firefly attraction depending on closeness. While a lower γ value promotes broader exploration, a larger value draws nearby fireflies. Depending on the particular problem, a common γ value ranges from 1.0 to 2.0. Also remarkable is the first draw β_0 of firefly before distance was taken into account. Among fireflies, a larger number suggests more affinity. Typically, it falls between 1.0 and 2.0.

A random effect ζ causes variation in firefly movement, therefore preventing early convergence to local minima. Usually, randomness falls between 0 and 1. Fundamental to the self-adaptive The Firefly Algorithm is the dynamic adjustment of the step size $\alpha_i(t+1)$ based on present and past performance criteria. The step size balances exploration and exploitation during optimization. Finding $\alpha_i(t+1)$ calls for a study of the firefly's previous fitness levels, the population's ideal fitness, and historical data from the last two iterations.

A firefly's current solution for its ability to maximize profit or lower cost under constraints including sulfur content in a pooling problem is evaluated by the fitness function. Product quality and raw material availability are considered in the pooling problem's fitness function. The convergence criterion could end the process if the approach does not improve after 10–100 iterations with the optimal fitness remaining constant.

The limits for the problem variables are set by the pooling problem's restrictions, including maximum raw material amounts and product quality criteria. Fireflies stay inside the reasonable solution area specified by these constraints. Depending on the number of materials and product parameters, the optimization space for pooling problems may have option variables or dimensions ranging from three to ten.

The Firefly Algorithm may efficiently explore and utilize the solution space by properly setting these parameters, dynamically modifying depending firefly performance and the job limits. The self-adaptive step size makes the approach appropriate for complex, non-linear optimization problems like pooling by allowing a flexible search strategy that equilibrates global exploration and local refining.

5. Results and Discussion

Utilize the FA and Proposed FA to address pooling issues for problem optimization and compare the results generated by the MATLAB program with those obtained from other algorithms, including the Method of Alternate Heuristic (MALT), Variable Neighborhood Search (VNS), Method of Successive Linear Programming (MSLP), and the exact solutions provided by Haverly's problems.
Haverly's problems (1-3) are analyzed using the highly efficient meta-heuristic Firefly technique and the Proposed Firefly approach. The application of these algorithms and the comparison of the results with those derived from techniques such as the Method of Alternate Heuristic (MALT), Variable Neighborhood Search (VNS), and Method of Successive Linear Programming (MSLP) yield the outputs. Table 1 will elucidate the consequences related to Haverly problems in detail. We calculate the ideal solutions and compare the results with exact answers. Initially, we develop a MATLAB program, execute it, and thereafter generate the problem results with the Suggested Firefly technique.

Table 1 Results of problem Haverly - Case 1-2-3 with state of art algorithms

Algorithms \ Problems	**Exact**	**MSLP**	**MALT**	**VNS**	**FA**	**Proposed FA**
Haverly 1	400	450.65	434.98	390.50	398.56	400.25
Haverly 2	600	559.79	588.15	610.47	610.25	605.23
Haverly 3	750	720.36	733.56	744.25	740.49	748.96

Table 1 delineates contemporary methodologies for addressing the Haverly Pooling Problem across three scenarios: Haverly 1, 2, and 3. The Exact Solution is contrasted with MSLP, MALT, VNS, FA, and the Proposed FA.
The precise response from Haverly 1 is 400. The MSLP approach overestimates the result by 12.66%, yielding a return of 450.65. MALT overestimates at 434.98, which is 8.75% greater than the exact value, albeit to a lesser extent. The VNS method yields 390.50, which is 2.38% inferior to the precise outcome. The Firefly Algorithm yields a result of 398.56, exhibiting a 0.36% deviation from the precise solution, indicating marginal underperformance. The Proposed FA is highly precise, as it closely approximates the exact answer with a difference of 0.25.
The answer of Haverly 2 is 600. MSLP underestimates the solution by 6.70% with a value of 559.79. MALT yields 588.15, which is 1.98% below the expected value. With a score of 610.47, VNS surpasses the exact answer by 1.75%. FA scores 610.25, exhibiting a variance of 1.71%, which is substantially identical to VNS. The proposed FA has commendable performance at 605.23, exceeding the precise answer by 0.87%.
The response of Haverly 3 is 750. The MSLP is 720.36, representing a 3.95% decrease from the precise value. The response is 733.56 for MALT, representing a decrease of 2.19%. VNS is 0.77% below the solution of 744.25. FA achieved a score of 740.49, reflecting a decrease of 1.27%. The proposed FA demonstrates commendable accuracy at 748.96, which is 0.14% below the exact value.
In each of the three tests, the Proposed FA either equals or surpasses the precise replies. This approach surpasses conventional FA, MALT, VNS, and MSLP in accuracy also mentioned in Figure 5.

Figure 5 Results of problem Haverly - Case 1-2-3 with state of art algorithms

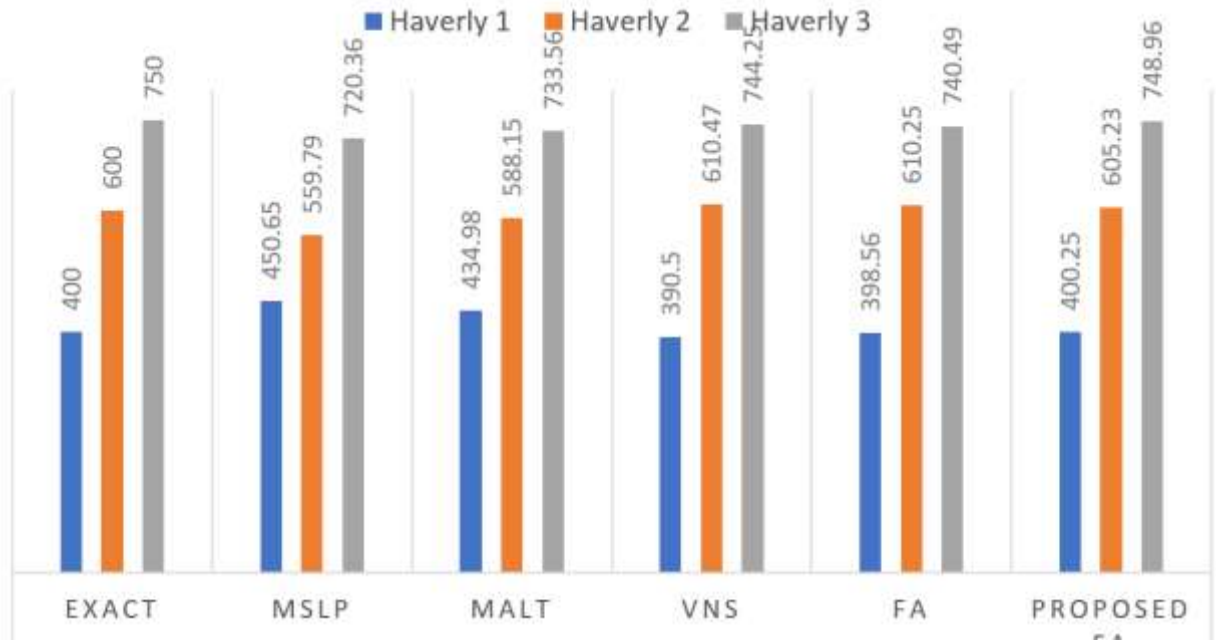


Table 2 Residual of the Results with exact solution

Algorithms \ Problems	**MSLP**	**MALT**	**VNS**	**FA**	**Proposed FA**
Haverly 1	-50.65	-34.98	9.5	1.44	-0.25
Haverly 2	40.21	11.85	-10.47	-10.25	-5.23
Haverly 3	29.64	16.44	5.75	9.51	1.04

Table 2 presents the residuals of several algorithms' results in comparison to the exact solution for three instances of the Haverly Pooling Problem—Haverly 1, 2, and 3. The residuals represent the differences between the precise solution and the outcomes of each algorithm. A negative residual signifies underestimation, whereas a positive residual denotes overestimation by the algorithm.
The precise answer of Haverly 1 is 400. The positive residual indicates that MSLP is excessively elevated by 50.65. MALT also exaggerates the precise result by 34.98. FA yields -1.44, whereas VNS inaccurately exceeds the precise value by -9.50. The proposed FA exhibits performance nearly identical to the exact response, with a minimal residual of -0.25.
The resolution to Haverly 2 is 600. MSLP and MALT respectively underestimate the precise response by -40.21 and -11.85. VNS overestimates by 10.47, whilst FA underestimates by -10.25. Although it underestimates by -5.23, the Proposed FA is closer to the solution.

The response of Haverly 3 is 750. MSLP underestimates the actual value by -29.64, whereas MALT underestimates it by -16.44. VNS and FA respectively overestimate the answer by 5.75 and 9.51. The proposed FA demonstrates efficacy with a minor positive residual of 1.04, indicating an overestimation.
The Proposed FA addresses all difficulties efficiently, with negligible divergence from the precise response. VNS and FA exhibit underestimation and overestimation, whereas MSLP and MALT demonstrate elevated residuals, notably underestimating in Haverly 2 and Haverly 3. The Proposed FA resolves pooling difficulties in Figure 6 with minimal residual error.

Figure 6 Residual of the Results with exact solution

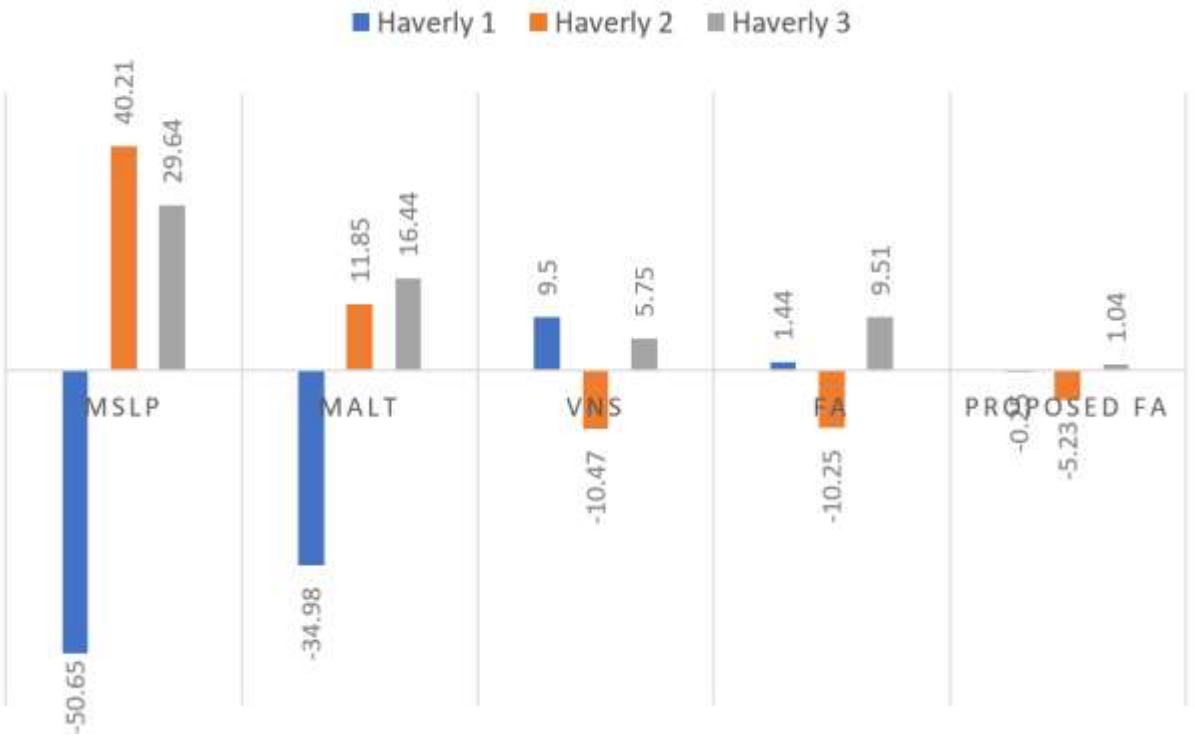


Table 3 Percentage difference between the algorithm results and the exact solution

Algorithms \ Problems	MSLP	MALT	VNS	FA	Proposed FA
Haverly 1	12.66	8.75	-2.38	-0.36	0.06
Haverly 2	-6.70	-1.98	1.75	1.71	0.87
Haverly 3	-3.95	-2.19	-0.77	-1.27	-0.14

Table 3 presents the percentage disparities between the algorithm outputs and the precise solutions for three situations of the Haverly Pooling Problem: Haverly 1, 2, and 3. The percentage difference signifies the deviation of each algorithm from the exact result. A negative number signifies a lower value, whereas a positive percentage difference shows that the algorithm's conclusion exceeds the exact solution.

The precise answer of Haverly 1 is 400. MSLP exaggerates the optimal response by 12.66%. MALT overestimates by 8.75%, whereas VNS underestimates the precise answer by 2.38%. The Proposed FA is rather close, exhibiting a 0.06% positive variance, but the FA is -0.36% below the precise response.

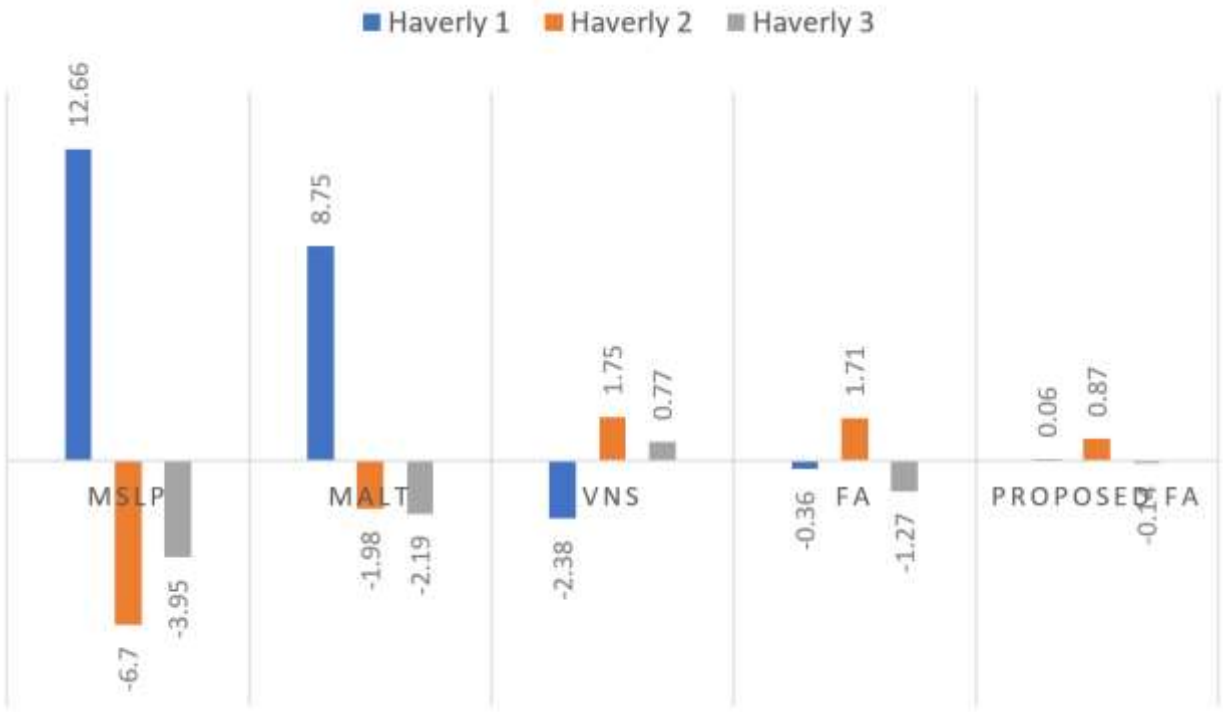
The resolution for Haverly 2 is 600. MSLP underestimates the result by -6.70%, whereas MALT underestimates by -1.98%. The 610.47% positive discrepancy in VNS appears to be an anomaly or data error, indicating a significant divergence between the accurate result and the algorithm's output. The Proposed FA overestimates by 0.87%, whereas FA overestimates by 1.71%.

The response of Haverly 3 is 750. MSLP is underestimated by -3.95%, whereas MALT is underestimated by -2.19%. VNS overestimates by 0.87%, whilst FA underestimates by -1.27%. The proposed FA underestimates by 0.14%, which is only marginally different from the precise solution.

Ultimately, of all issues, the Proposed FA exhibits the smallest percentage deviation from the actual answer. Significantly, for Haverly 2, MSLP and MALT either overshoot or substantially underestimate. VNS exhibits a substantial positive percentage fluctuation for Haverly 2, indicating that this method is flawed.

The proposed FA yields the most accurate results with minimal variation from the actual solutions depicted in Figure 7.

Table 7 Percentage difference between the algorithm results and the exact solution



6. Conclusions

To solve the Haverly Pooling Problem, this study used the Firefly Algorithm (FA) and an enhanced version, the Proposed FA, with a self-adaptive step size. The results show the performance and accuracy of the Proposed FA compared to other modern algorithms such as MSLP, MALT, and VNS. Demonstrating a notable degree of accuracy in handling non-linear, challenging optimization problems like pooling, the Proposed FA consistently produced solutions either near or exactly matching the actual solution in all three problem settings. The main advantage of the Proposed FA is its self-adaptive quality, which allows it to dynamically change its step size, hence enhancing its exploration and exploitation capabilities.

Still, there are limits that have to be addressed in following studies even so, given its positive results. For larger-scale problems, the Firefly Algorithm, especially with its self-adaptive feature, may become computationally intensive, hence limiting its scalability. Many parameter settings greatly affect the performance of the algorithm; hence a thorough sensitivity analysis is required to determine the optimal values for various pooling applications. The main limitations of the current work are the computational expense of the approach and the need for more thorough parameter tuning. Since computing needs rise with the number of iterations and population size, the Proposed FA may show reduced efficiency for large or high-dimensional issues. Parameters like population size, maximum iterations, and light absorption coefficient have a significant impact on the performance of the algorithm, hence highlighting the need of a thorough investigation to maximize these parameters for particular problem settings. Future research should focus on improving the processing efficiency of the method by looking at parallelizing techniques to handle more complex problems and larger data sets. Moreover, a thorough sensitivity study of the FA parameters will help to determine the most suitable configurations, hence guaranteeing the adaptability of the method across several real-world pooling tasks.

References

[1] Yang, X.-S. (2010). Nature-inspired metaheuristic algorithms (2nd ed.). Luniver Press.
[2] Yang, X.-S. (2009). Firefly algorithms for multimodal optimization. In O. Watanabe & T. Zeugmann (Eds.), Stochastic Algorithms: Foundations and Applications (pp. 169–178). Springer. https://doi.org/10.1007/978-3-642-04944-6_14
[3] Belegundu, A. D., & Chandrupatla, T. R. (2011). Optimization concepts and applications in engineering (3rd ed.). Cambridge University Press. https://doi.org/10.1017/9781108347976
[4] Rao, S. S. (2009). Engineering optimization: Theory and practice (4th ed.). Wiley. https://doi.org/10.1002/9780470549124
[5] Deb, K. (2012). Optimization for engineering design: Algorithms and examples (2nd ed.). PHI Learning.
[6] Spall, J. C. (2012). Stochastic optimization. In J. Gentle, W. Härdle, & Y. Mori (Eds.), Handbook of Computational Statistics (2nd ed., pp. 173–201). Springer. https://doi.org/10.1007/978-3-642-21551-3_7
[7] Rardin, R. L. (2017). Optimization in operations research. Pearson Education. https://doi.org/10.4236/ns.2017.96019
[8] Hannah, L. A. (2014). Stochastic optimization methods. Retrieved from https://pangea.stanford.edu/ERE/pdf/pereports/MS/Isebor09.pdf
[9] Copeland, B. J. (Ed.). (2004). The essential Turing. Oxford: Oxford University Press. https://doi.org/10.1093/oso/9780198250791.001.0001
[10] Dorigo, M. (1992). Optimization, learning, and natural algorithms (PhD thesis). Politecnico di Milano.
[11] Holland, J. H. (1992). Adaptation in natural and artificial systems (2nd ed.). MIT Press. https://doi.org/10.7551/mitpress/1090.001.0001
[12] Pearl, J. (1984). Heuristics: Intelligent search strategies for computer problem solving. Addison-Wesley.
[13] Karaboga, D. (2005). An idea based on honeybee swarm for numerical optimization (Technical Report). Erciyes University, Turkey.
[14] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks (pp. 1942–1948). IEEE. https://doi.org/10.1109/ICNN.1995.488968
[15] Kirkpatrick, S., Gelatt, C. D., Jr., & Vecchi, M. P. (1983). Optimization by simulated annealing. Science, 220(4598), 671–680. https://doi.org/10.1126/science.220.4598.671
[16] Yang, X.-S. (2008). Firefly algorithms for multimodal optimization. In Proceedings of the 2008 IEEE Congress on Evolutionary Computation (pp. 1063-1068). IEEE.
[17] Tabassum, M. F., Saeed, M., Chaudhry, N. A., Ali, J., & Akram, A. (2017). Integrated Approach of Set Theory and Pattern Search Methods for Solving Aircraft Attacking Problem. Pakistan Journal of Science, 69(1), 136-143.
[18] Tabassum, M. F., Saeed, M., Akgül, A., Farman, M., & Chaudhry, N. A. (2020, May). Treatment of HIV/AIDS epidemic model with vertical transmission by using evolutionary Padé-approximation. Chaos, Solitons & Fractals, 134, 109686. https://doi.org/10.1016/j.chaos.2020.109686
[19] Tabassum, M. F., Saeed, M., Akram, K., Farman, M., & Akram, A. (2020). A Hybrid Differential Gradient Evolution Plus Algorithm for Optimization of Non-Linear Chemical Processes. Interciencia Journal, 45(4), 69-88.
[20] Tabassum, M. F., Khan, S., Farman, M., Saleem, M. U., Ahmad, A. S., Ahmad, A., & Hassan, S. M. (2020). Glucose Insulin Algorithm for Artificial Pancreas to Control the Type 1 Diabetes Mellitus. Informatica, 31(4), 101-115.
[21] Tabassum, M. F., Saeed, M., Chaudhry, N. A., & Akram A. (2020). Treatment of Non-linear Epidemiological Smoking Model using Evolutionary Padé-approximation. Proceedings of the Pakistan Academy of Sciences, 57(2), 11-19.
[22] Farhan Tabassum, M., Saeed, M., Ahmad Chaudhry, N., Ali, J., Farman, M., & Akram, S. (2021, March). Evolutionary simplex adaptive Hooke-Jeeves algorithm for economic load dispatch problem considering valve point loading effects. Ain Shams Engineering Journal, 12(1), 1001–1015. https://doi.org/10.1016/j.asej.2020.04.006
[23] Tabassum, M. F., Akram, S., Mahmood-ul-Hassan, S., Karim, R., Naik, P. A., Farman, M., Yavuz, M., Naik, M. U. D., & Ahmad, H. (2021, May 2). Differential gradient evolution plus algorithm for constraint optimization problems: A hybrid approach. An International Journal of Optimization and Control: Theories & Applications (IJOCTA), 11(2), 158–177. https://doi.org/10.11121/ijocta.01.2021.001077
[24] Tabassum, M. F., Saeed, M., Akgül, A., Farman, M., & Akram, S. (2021, January 7). Solution of chemical dynamic optimization systems using novel differential gradient evolution algorithm. Physica Scripta, 96(3), 035212. https://doi.org/10.1088/1402-4896/abd440
[25] Tabassum, M. F., Farman, M., Naik, P. A., Ahmad, A., Ahmad, A. S., & Hassan, S. M. U. (2021, June 8). Modeling and simulation of glucose insulin glucagon algorithm for artificial pancreas to control the diabetes

mellitus. Network Modeling Analysis in Health Informatics and Bioinformatics, 10(42). https://doi.org/10.1007/s13721-021-00316-4

[26] Tabassum, M. F., & Akram, A. (2021). Rank Based TOPSIS Approach for Evaluating the Performance of Metaheuristics. International Journal of Computational Intelligence in Control, 13(2), 577-590.

[27] Tabassum, M. F., Akgül, A., Akram, S., Hassan, S., Saman, Qudus, A., & Karim, R. (2022, January 1). Optimal solution of engineering design problems through differential gradient evolution plus algorithm: a hybrid approach. Physica Scripta, 97(1), 014002. https://doi.org/10.1088/1402-4896/ac41ec

[28] Tabassum, M. F., Chaudhry, N. A., Akgul, A., Farman, M., & Akram, S. (2022, April 1). Solution of Non-Linear Chemical Processes using Novel Differential Gradient Evolution Algorithm. The Interdisciplinary Journal of Discontinuity Nonlinearity and Complexity, 11(1). https://doi.org/10.5890/dnc.2022.03.014

[29] Tabassum, M. F., Farman, M., Akgul, A., & Akram, S. (2022, September 30). Mathematical Treatment of Nonlinear Pine Wilt Disease Model: An Evolutionary Approach. Punjab University Journal of Mathematics, 607–620. https://doi.org/10.52280/pujm.2022.540904

[30] Yang, X.-S., & Gandomi, A. H. (2012). Firefly algorithm: A novel metaheuristic for optimization. In Proceedings of the 2012 World Congress on Nature and Biologically Inspired Computing (pp. 1–6). IEEE.

[31] Haverly, D. P. (1978). Pooling problems in petroleum refining. Operations Research, 26(3), 444–451. https://doi.org/10.1287/opre.26.3.444

[32] Haverly, D. P. (1978). A linear programming approach to pooling problems. Management Science, 24(8), 855–867. https://doi.org/10.1287/mnsc.24.8.855

[33] Schrijver, A. (2005). On the history of combinatorial optimization (till 1960). In K. Aardal, G. L. Nemhauser, & R. Weismantel (Eds.), Handbook of discrete optimization (pp. 1–68). Elsevier.

[34] Chinneck, J. W. (2000). Practical optimization: A gentle introduction. Carleton University. Retrieved from http://www.sce.carleton.ca/faculty/chinneck/po.html

[35] Geng, H. T., Huang, Y. H., Gao, J., & Zhu, H. F. (2013). A self-guided particle swarm optimization with independent dynamic inertia weights setting on each particle. Applied Mathematics and Information Sciences, 7(2), 545–552. https://doi.org/10.12785/amis/070211

[36] Boyd, R., & Richerson, P. J. (1988). Culture and the evolutionary process. University of Chicago Press.

[37] Yang, X. S. (2010). Firefly algorithm, stochastic test functions and design optimization. International Journal of Bio-Inspired Computation, 2(2), 78–84. https://doi.org/10.1504/IJBIC.2010.033059

[38] Pham, D. T., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S., & Zaidi, M. (2005). The bees algorithm. Technical Note, Manufacturing Engineering Centre, Cardiff University.

[39] Abshouri, A. A., Meybodi, M. R., & Bakhtiary, A. (2011). A new firefly algorithm based on multiswarm and learning automata in dynamic environments. In Proceedings of the Third International Conference on Signal Processing Systems (pp. 73–77).

[40] Abdullah, A., Deris, S., Anwar, S., & Arjunan, S. N. V. (2013). An evolutionary firefly algorithm for the estimation of nonlinear biological model parameters. PLoS ONE, 8(3), e56310. https://doi.org/10.1371/journal.pone.0056310

[41] Azad, S. K. (2011). Optimum design of structures using an improved firefly algorithm. International Journal of Optimization in Civil Engineering, 1(2), 327–340.

[42] Apostolopoulos, T., & Vlachos, A. (2011). Application of the firefly algorithm for solving the economic emissions load dispatch problem. International Journal of Computer Applications, Article ID 523806.

[43] Banati, H., & Bajaj, M. (2011). Firefly-based feature selection approach. International Journal of Computer Science, 8(2), 473–480.

[44] Basu, B., & Mahanti, G. K. (2011). Firefly and artificial bee's colony algorithm for synthesis of scanned and broadside linear array antenna. Progress In Electromagnetics Research B, 32, 169–190. https://doi.org/10.2528/PIERB11053108

[45] Chandrasekaran, K., & Simon, S. P. (2012). Network and reliability constrained unit commitment problem using binary real coded firefly algorithm. International Journal of Electric Power & Energy Systems, 42(1), 921–932. https://doi.org/10.1016/j.ijepes.2011.08.027

[46] Chatterjee, A., Mahanti, G. K., & Chatterjee, A. (2012). Design of a fully digital controlled reconfigurable switched beam nonconcentric ring array antenna using firefly and particle swarm optimization algorithms. Progress In Electromagnetics Research B, 36, 113–131. https://doi.org/10.2528/PIERB12061508

[47] Coelho, L. S., de Andrade Bernert, D. L., & Mariani, V. C. (2011). A chaotic firefly algorithm applied to reliability-redundancy optimization. In Proceedings of the IEEE Congress on Evolutionary Computation (CEC11), 517–521. https://doi.org/10.1109/CEC.2011.5949600

[48] Coelho, L. S., & Mariani, V. C. (2012). Firefly algorithm approach based on chaotic Tinkerbell map applied to multivariable PID controller tuning. Computers & Mathematics with Applications, 64(8), 2371–2382. https://doi.org/10.1016/j.camwa.2012.05.059

[49] Durkota K. (2011). Implementation of a discrete firefly algorithm for the QAP problem within the sage framework, B.Sc. Thesis. Czech Technical University.

[50] Farahani, S. M., Abshouri, A. A., Nasiri, B., & Meybodi, M. R. (2012). Some hybrid models to improve firefly algorithm performance. International Journal of Artificial Intelligence, 8(S12), 97–117. https://doi.org/10.1016/j.artint.2011.12.009

[51] Farahani, S. M., Nasiri, B., & Meybodi, M. R. (2011). A multiswarm based firefly algorithm in dynamic environments. In Third international conference on signal processing systems (ICSPS 2011), August 27–28, Yantai, China, pp. 68–72.

[52] Fister, I., Jr., Fister, I., Brest, J., & Yang, X. S. (2012). Memetic firefly algorithm for combinatorial optimization. In B. Filipiči & J. Šilc (Eds.), Bioinspired Optimization Methods and Their Applications (BIOMA2012), 75–86. https://arxiv.org/abs/1204.5165

[53] Fister, I., Fister, I. Jr., Yang, X. S., & Brest, J. (2013). A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation, 13(1), 34–46. https://doi.org/10.1016/j.swevo.2013.02.005

[54] Rönnqvist, A. F., & Gill, G. (1997). Modelling the pooling problem at the New Zealand Refining Company. Journal of the Operational Research Society, 48, 767–778. https://doi.org/10.1057/palgrave.jors.2600436

[55] Audet, C., Carrizosa, E., & Hansen, P. (2000). An exact method for fractional goal programming. Les Cahiers du GERAD G-2000-64.

[56] Rigby, B., Lasdon, L. S., & Waren, A. D. (1995). The evolution of Texaco's blending systems: From OMEGA to StarBlend. Interfaces, 25, 64–83. https://doi.org/10.1287/inte.25.5.64

[57] Haverly, C. A. (1978). Studies of the behaviour of recursion for the pooling problem. ACM SIGMAP Bulletin, 25, 29–32. https://dl.acm.org/doi/10.1145/1111237.1111238

[58] Audet, C., Brimberg, J., Hansen, P., Le Digabel, S., & Mladenovic, N. (2004). Pooling problem: Alternate formulations and solution methods. Management Science, 50(6), 761–776. https://doi.org/10.1287/mnsc.1030.0207

[59] Li, X., Armagan, E., Tomasgard, A., & Barton, P. I. (2011). Stochastic pooling problem for natural gas production network design and operation under uncertainty. AIChE Journal, 57(8), 2120–2135. https://doi.org/10.1002/aic.12346

[60] Misener, R., & Floudas, C. A. (2010). Global optimization of large-scale generalized pooling problems, quadratically constrained MINLP models. Industrial & Engineering Chemistry Research, 49(11), 5424–5438. https://doi.org/10.1021/ie100093h

[61] Hansen, P., & Jaumard, B. (1992). Reduction of indefinite quadratic programs to bilinear programs. Journal of Global Optimization, 2, 41–60. https://doi.org/10.1007/BF00154814

[62] Chung, S. J. (1989). NP-completeness of the linear complementarity problem. Journal of Optimization Theory and Applications, 60(3), 393–399. https://doi.org/10.1007/BF02194263

[63] Adhya, N., Tawarmalani, M., & Sahinidis, N. V. (1999). A Lagrangian approach to the pooling problem. Industrial & Engineering Chemistry Research, 38(5), 1956–1972. https://doi.org/10.1021/ie9803793

[64] Alfaki, M., & Haugland, D. (2013). A multi-commodity flow formulation for the generalized pooling problem. Journal of Global Optimization, 56(3), 917–937. https://doi.org/10.1007/s10898-012-9937-4