Application of Modified Flower Pollination Algorithm on Mechanical Engineering Design Problem

Ong Kok Meng, Ong Pauline*, Sia Chee Kiong, Hanani Abdul Wahab, Noormaziah Jafferi
ongkokmeng93@hotmail.com, {ongp*, sia, nani, normazia}@uthm.edu.my

Abstract. The aim of the optimization is to obtain the best solution among other solutions in order to achieve the objective of the problem without evaluation on all possible solutions. In this study, an improved flower pollination algorithm, namely, the Modified Flower Pollination Algorithms (MFPA) is developed. Comprising of the elements of chaos theory, frog leaping local search and adaptive inertia weight, the performance of MFPA is evaluated in optimizing five benchmark mechanical engineering design problems - tubular column design, speed reducer, gear train, tension/compression spring design and pressure vessel. The obtained results are listed and compared with the results of the other state-of-art algorithms. Assessment shows that the MFPA gives promising result in finding the optimal design for all considered mechanical engineering problems.

1. Introduction
Optimization is numerical methods which search and identify the best candidate from a collection of alternatives without evaluation on all possible alternatives [2]. In engineering field, engineers are often required to build the smallest heat exchanger that can achieve the desired heat transfer or to build the lowest cost engine for a car. In this regard, engineer will design a product by finding the best combination of parameters using optimization technique that will satisfy human needs, and this process is known as engineering design optimization. Therefore, engineering design optimization is very important to solve real world problem by finding the best solution with the least work.

To initiate the optimization solution procedure, the engineering design problem is derived in mathematical form, consisting of the objective function, and probably with set of constraints. The objective function is then solved by using optimization algorithm in order to find its optimal variables, which either maximizes or minimizes the underlying problem. For instance, optimal variables obtained from the optimization algorithm will give the objective function the minimum value for the case study of designing lowest cost engine for a car.

Most of the optimization problems can be mathematically written as:

\[
\begin{align*}
\text{Minimize} & \quad f_i(x), & (i = 1, 2, ..., M), \\
\text{subject to} & \quad \emptyset_j(x) = 0, & (j = 1, 2, ..., J), \\
& \quad \psi_k(x) \leq 0, & (k = 1, 2, ..., K),
\end{align*}
\]

Where \(f_i(x), \emptyset_j(x)\) and \(\psi_k(x)\) are functions of the design variables, \(x = (x_1, x_2, ..., x_n)^T\). \(f_i(x)\) is known as objective function and the design variables can be real continuous, discrete or the mixed of them. Based on optimization equations in Equation (1), if \(J=K=0\), there is no constraint for that...
particular optimization problem. For optimization problems which are classified as equality-constrained problem, their \( K=0 \) and \( J \geq 1 \), meanwhile for those which are classified as inequality-constrained problem, their \( J=0 \) and \( K \geq 1 \) [3].

In order to solve the optimization problem, the development of robust algorithms in solving complex optimization problems initiated since 1960s, where the differential calculus methods form its foundation. The invention of high-speed digital computers in the middle of the twentieth results in the arising of new areas in optimization theory - the bio-inspired algorithm, offers an important breakthrough in optimization. Since then, the bio-inspired optimization algorithm has been the in thing.

Nature has evolved more than millions of years and she has found her best solution in overcoming problems that she has met. The success of how nature finds its way in solving problems triggers the formulation of bio-inspired algorithms [3]. Exemplary, the genetic algorithm (GA), perhaps, one of the most widely used bio-inspired techniques, is a model of biological evolution based on the Charles Darwin’s theory [4]. The particle swarm optimization (PSO) algorithm, on the other hand, imitates the behavior of social organisms. The manner of how each particle communicates and shares the information to others forms the basis of PSO [5].

The success of GA and PSO in solving complex optimization problems prompts the formulation of more bio-inspired optimization algorithms, including the flower pollination algorithm (FPA) [6]. In nature, flower pollination is a marvelous process where the evolutionary objective such as reproduction is maximized efficiently. There are two types of pollination - abiotic and biotic, in which the FPA is derived based on such fascinating characteristics. The pioneer work of FPA shows its superiority over GA and PSO in the considered benchmark functions [6], but with fewer control parameters. The comprehensive ability for optimization, simplicity, and yet rapid convergence of FPA have gained widespread implementation in diverse areas; including optimization of wireless sensor network life time [7], retinal vessel segmentation [8], designing compensator damping controller [9], and allocation of capacitors [10].

This paper presents a modified flower pollination algorithm (MFPA), attempting to optimize five benchmark Mechanical Engineering design problems.

The paper is organized as follows. The basic idea of flower pollination process which leads to the formulation of FPA is first introduced. Subsequently, the chaotic maps, frog leaping local search and inertia weight are introduced as well as the procedure of proposed MFPA are presented. The performance of the MFPA is then evaluated on five benchmark and the results are compared with the results of the other state-of-art algorithm.

2. Flower Pollination Algorithm
There are between 5% and 20% of the flowering plants around 105 million years ago, but it increases to 80% after 40 million years. Scientists believe that the flowering plants start to dominate the landscape 125 million years ago during the Cretaceous Period. The mystery and yet amazing evolution of flowering plant drives Xin-She Yang to deep in his studies and develop FPA for global optimization [6].

The main factor causes the evolution of flowering plant is reproduction via pollination. Pollination can be categorized into two types - abiotic and biotic. Abiotic pollination does not require any pollinators to transfer the pollen. It depends on wind and water to transfer the pollen from one flower to another, known as wind pollination and water pollination, respectively. Meanwhile, biotic pollination requires pollinators to transfer the pollen. The examples of pollinator are beetle, fly, bee, butterfly, moth, bird and bat [11]. Furthermore, there are two types of pollination mechanism, specifically, self-pollination and cross-pollination. Self-pollination occurs when pollen is transferred from one flower to another same species flower, whereas cross-pollination occurs when pollen is transferred from one flower to another different species flower [11].

The aforesaid characteristics of flower pollination process are translated into the FPA by Xin-She Yang [6]. In the FPA, the biotic and cross-pollination are considered as global searching, as it has high
probability occurring at long distance. In addition, the pollinators such as bees, birds and bats might behave as Lévy flight behavior which obey a Lévy distribution [12]. Meanwhile, the abiotic and self-pollination are considered as local searching in the FPA, as it may occur at short distance only. For simplicity, the developed FPA is idealized based on the assumptions of [6]:

1) Pollen-carrying pollinators perform the pollination using Lévy flight, where such biotic and crosspollination are considered as global searching.
2) Abiotic and self-pollination are equivalent to local searching.
3) Flower constancy is equivalent to a reproduction probability. It is proportional to the similarity of two flowers involved.
4) The switch probability, \( p \in [0,1] \) controls both local and global pollination. Local pollination has significant fraction \( p \) and dominates overall pollination activities due to physical proximity and wind.

A set of location updating formula are required to implement these idealized rules. First, for global pollination process, the pollinators such as bee carried the flower pollens and travelled over a long distance (Rule 1). This pollination process can be expressed mathematically as:

\[
x_{i}^{t+1} = x_{i}^{t} + L(x_{i}^{t} - g_{*})
\]  (2)

Where \( x_{i}^{t} \) is the pollen i or solution vector \( x_{i} \) at iteration \( t \), and \( g_{*} \) is the current best solution found among all solutions at current iteration. The Lévy flight is used to imitate the flying pattern of the insect [13]. Therefore, \( L>0 \) is drawn from a Lévy distribution

\[
L \sim \frac{\Gamma(\lambda) \sin(\frac{\pi \lambda}{2})}{\pi} \frac{1}{s^{1+\lambda}}, \quad (s \gg s_{0} \gg 0)
\]  (3)

Where \( \Gamma(\lambda) \) is the standard gamma function. This distribution is valid for large steps \( s>0 \).

Considering the flower constancy, \( \epsilon \) the local pollination process (Rule 2) can be represented mathematically as:

\[
x_{i}^{t+1} = x_{i}^{t} + \epsilon(x_{j}^{t} - x_{k}^{t})
\]  (4)

Where \( x_{i}^{t} \) and \( x_{j}^{t} \) represent pollen from different flowers of the same species. The use of flower constancy \( \epsilon \) in Rule 3 ensures that the same species of flowers (similar solutions) are chosen more frequently and thus encouraging the exploitation. According to Rule 4, switch probability is bias towards local pollination. In the pioneer work of the FPA, \( p=0.8 \) works well for most applications.

3. Chaotic Maps

Chaos is a random state found in the non-linear dynamical deterministic system, possesses non-period, non-converging and bounded properties [14]. The use of chaotic sequences is more beneficial than the random sequences due to its non-repetition and ergodicity properties. Borrowing the advantages of ergodicity, non-repetition and randomness of the chaotic sequences, the chaotic map is replacing the random sequences in generating the initial population in the FPA in this study. This is to ensure that the diversity of the initial population can be improved, where the distribution of the initial population is more uniform. Ten different chaotic maps are listed in Table 1 and circle map is selected for the integration with FPA.
Table-1: Ten different chaotic maps

| Name            | Formula                                                                 |
|-----------------|-------------------------------------------------------------------------|
| Chebyshev map   | $x_{n+1} = \cos^{-1}(n \times \arccos(x_n))$                           |
| Circle map      | $x_{n+1} = (x_n + 0.2 - (0.5/2\pi)\sin(2\pi x_n)) \times \text{mod}(1)$ |
| Gauss map       | $x_{n+1} = \begin{cases} 0, & x_n = 0 \\ (1/x_n) \times \text{mod}(1), & x_n \in (0,1) \end{cases}$ |
| Iterative map   | $x_{n+1} = \sin(0.7\pi/x_n)$                                           |
| Logistic map    | $x_{n+1} = 4x_n(1 - x_n)$                                               |
| Piecewise map   | $x_{n+1} = \begin{cases} 2.5x_n, & x_n \in (0,0.4) \\ (x_n - 0.4)/0.1, & x_n \in (0.4,0.5) \\ (0.6 - x_n)/0.1, & x_n \in (0.5,0.6) \\ (1 - x_n)/0.4, & x_n \in (0.6,1.0) \end{cases}$ |
| Sine map        | $x_{n+1} = \sin(x_n \times \pi)$                                        |
| Singer map      | $x_{n+1} = 1.07(7.86x_n - 23.31(x_n)^2 + 28.75(x_n)^3 - 13.302875(x_n)^4)$ |
| Sinusoidal map  | $x_{n+1} = 2.3(x_n)^2 \times \sin(x_n \times \pi)$                      |
| Tent map        | $x_{n+1} = \begin{cases} x_n/0.7, & x_n < 0.7 \\ (10/3) \times (1 - x_n), & x_n \geq 0.7 \end{cases}$ |

4. Frog Leaping Local Search

The Shuffled Frog-Leaping Algorithm (SFLA) was proposed by Eusuff, Lansey and Pasha in 2003. SFLA is classified as memetic metaheuristic algorithm. The creation of SFLA is inspired by the foraging behaviours of frogs.

First, a population size of frog is determined and stayed in a wetland (search space). Each frog in the wetland hold the information about the amount of food (fitness) in that location (solution) within the wetland. It is then divided into a number of groups of frogs known as number of memeplexes.

In each group, the frog that holds the least amount of food (worst fitness) is moved to new location in wetland due to information sharing from the frog that holds the most amount of food (best fitness) in that particular group. If the frog that holds the least amount of food is moved to a location where the amount of food is lesser, means no improvement, the frog will then randomly move to another new location in wetland. This process is repeated in each group and it is known as the iteration within each memeplex.

When the process undergone by each group of frogs happened for a period of time, all groups of frogs are gathered and joined as one population size for information sharing among those frogs. If the largest amount of food (convergence criteria) found by the frog in that population size is unsatisfied, the population size of frog is then divided into a number of groups of frogs again and continually searching for food until they find the largest amount of food in that wetland. The flowchart of SFLA is illustrated in Figure 1.

In SFLA, the position of the frog with the worst fitness in a group is changed due to equations as follows:
Change in frog position \((D_i) = \text{rand()} \cdot C \cdot (X_b - X_w)\) (5)

New position \(X_w = \text{current position } X_w + D_i, \ (D_{\text{max}} \gg D_i \gg -D_{\text{max}})\) (6)

where \(\text{rand()}\) a random number between 0 and 1, \(C\) is a search-acceleration factor, \(X_b\) is best fitness, \(X_w\) is worst fitness and \(D_{\text{max}}\) is the maximum allowed change in a frog’s position.

Figure-1: Flowchart of the SFLA

5. Inertia Weight

In basic FPA, the global searching of Levy flight is a random walk. Thus, there is no guaranteed on its fast convergence. To improve the performance of FPA in global searching, inertia weight is integrated into Levy flight. The formula of global searching of Levy flight in FPA is expressed in equation (2). The levy flight in equation (2) is used to imitate the fly pattern of the insect. To improve this formula, the Levy flight, \(L\) is multiplied by inertia weight, \(w\) as shown as follows:

\[
x_i^{t+1} = x_i^t + wL(x_i^t - g_*)
\]

\[
w = w_{\text{min}} \left(1 + \frac{w_{\text{max}}}{\sqrt{t}} \times \tanh \left(\frac{g_*}{g_1}\right)\right)
\]

Where \(g_*\) is the best fitness value in first iteration, \(t\) denotes the current iteration number and \(g_*\) is the obtained best fitness value in current iteration.

The inertia weight, \(w\) determines how far the new flower is located from the previous flower. As the number of iteration, \(t\) increases, the best fitness value, \(g_*\) in current iteration decreases for
minimization case. Wherefore, the value of \( w \) will be decreased from a larger value to a final value, \( w_{\min} \) at last. This indicates that the larger the difference of the new fitness of the current best flower from the fitness of the best flower in first iteration, the nearer the new location of flower will be shifted from previous location. To start with identifying the optimal value of \( w_{\min} \) and \( w_{\max} \), the value of \( w_{\min} \) and \( w_{\max} \) are set in the range of 1 to 10. Initially, \( w_{\min} \) is set to 1 whereas \( w_{\max} \) is set to 10 and then a parametric study is carried out to identify the most appropriate parameter. From the simulations, \( w_{\min} = 1 \) and \( w_{\max} = 8 \) works better for most applications. Thus, by integrating the inertia weight into Lévy flight, the search space exploration is improved more entirely and the solution can effectively move away from a local optima at the early phase and accelerate convergence at the later phase.

6. Modified Flower Pollination Algorithm

In the MFPA, the initial population is generated using the circle map, frog leaping local search is performed by each solution and when \( \text{rand}>\text{p} \), modified Levy flight with integration of inertia weight in global pollination is performed on that particular solution. The steps involved in the MFPA are as follows:

Step 1: Parameter Initialization
Initialize the relevant parameters of population size, \( n \), dimension of search space, \( d \), maximum iteration, \( \text{max}\_\text{iter} \), switch probability, \( p \), range of search space \([Lb, Ub]\), number of memplexes, \( m \) and iterations within each memplex, \( it \).

Step 2: Generate Initial Population using the best chaotic map
The \( n \) initial population is generated randomly, where \( \text{Pop} = (\text{flower}_1, \text{flower}_2, ..., \text{flower}_n) \), where \( \text{flower}_i = (x_{i1}, x_{i2}, ..., x_{id}) \), \( i = 1, ..., n \) and \( d \) is dimension of the search variables. Then, each dimension of the search variable in first flower, \( x_{i1} \) is randomly generated between zero and one where \( x_{i1} \in [0,1] \) and \( j = 1, ..., d \). Subsequently, each dimension of the search variable in the rest of the flower, \( x_{kj} \) is updated using the circle map as in Table 1 where \( k = 2, ..., n \). The generated chaotic sequences are further mapped within the solution space using \( X_{ij} = Lb + (Ub - Lb) x_{ij} \).

Step 3: Find the Best Solution
The fitness value of each solution is calculated and the best solution \( g^* \) is determined.

Step 4: Perform the Frog Leaping Local Search
For each solution, frog leaping local search is performed by following the flowchart in Figure 1.

Step 5: Perform the Global Search of Flower Pollination Algorithm
For each solution, an evenly distributed number, \( \text{rand} \) is generated from uniform distribution \([0,1]\). If \( \text{rand}>\text{p} \), global pollination is performed by using Equation (7), otherwise, the solution remain in its position.

Step 6: Update the Solution
The fitness value of each new solution is evaluated. The historical position is updated through comparison with the new solution. Subsequently, the best solution \( g^* \) is updated.

Step 7: Check Termination Condition
If the termination condition is satisfied, the optimal solution \( g^* \) is displayed. Otherwise, Step 4 is repeated.
7. Numerical Simulations

In order to validate the feasibility of the MFPA, the algorithm is tested by means of optimizing five benchmark Mechanical Engineering design problems. For each benchmark Mechanical Engineering design problem, MFPA is simulated for 30 trials. For all simulations, the switch probability, $p$, is assigned as 0.8 in accordance with the pioneer work.

Each Mechanical Engineering design problem will be begun with its objective function. Afterward, the design variables are listed with their boundaries. The constraint of those particular problems will be listed too if available. The information above will be encoded into MFPA and the algorithm will then solve this particular problem. Lastly, the result from solving each Mechanical Engineering design problem by using MFPA will be listed and compared with the result of the other state-of-art algorithm.

7.1. Tubular Column Design

A tubular section as shown in Figure 2 is designed to carry a compressive load $P = 2500 \text{ kgf}$ at minimum cost. Both ends of the tubular column are hinge joints. The properties of the material that made up this column are listed in Table 2.

![Figure-2: Tubular Column](image)

![Section A-A](image)

| Table-2: Properties of the Material |
|-------------------------------------|
| Yield Stress, $\sigma_y (kgf/cm^2)$ | 500    |
| Modulus of elasticity, $E (kgf/cm^2)$ | $0.85 \times 10^6$ |
| Weight density, $\rho (kgf/cm^3)$ | 0.0025 |

The length ($L$) of the column is 250cm. The stress induced in the column should be less than the buckling stress (constraint $g_1$) as well as the yield stress (constraint $g_2$). The mean diameter of the column is restricted to lie between 2 and 14 cm (design variable 1), and columns with thicknesses outside the range 0.2 to 0.8 cm (design variable 2) are not available in the market. Hence, the design variables in this problem are mean diameter of the column in centimetres, $d$, and thickness of the column, $t$. The cost of the column includes material and construction costs can be taken as $5W + 2d$, where $W$ is the weight in kilograms force [15].
The objective function of this model is given as following:

\[
\begin{align*}
    \text{Minimize } f(d, t) &= 5W + 2d \\
    &= 5\pi l d t + 2d \\
    &= 9.8d t + 2d
\end{align*}
\] (9)

Subject to

\[
\begin{align*}
    g_1 &= \frac{P}{\pi dt \sigma_y} - 1 \leq 0 \\
    g_2 &= \frac{8PL^2}{\pi^3 Edt (d^2 + t^2)} - 1 \leq 0
\end{align*}
\]

Where

\[
\begin{align*}
    2 \ll d \ll 14 \\
    0.2 \ll t \ll 0.8
\end{align*}
\]

Table 3 shows the results provided by MFPA in optimizing tubular column design problem. The population size, n of MFPA in optimizing this problem is 30 and maximum number of iteration for MFPA to optimizing this problem is 100. Table 3 also shows the result comparison between MFPA and Cuckoo Search algorithm (CS) [16]. Table 4 illustrates the comparison of the results reported by Hsu and Liu [17] as well as Rao [15] and MFPA. Bold sets in Table 4 are violated sets which they do not fulfill the condition of constraint.

| Table-3: Result Comparison between MFPA and CS |
|-----------------------------------------------|
| CS | MFPA |
| Mean diameter, d | 5.45139 | 5.4512 |
| Thickness, t | 0.29196 | 0.29197 |
| \( f_{\text{min}} \) | 26.50034 | 26.49995 |
| Maximum number of iteration | 15000 | 100 |
| Average elapsed time of each run (s) | 2.56919 | 0.36589 |

| Table-4: Best Solutions for the Tabular Column Example |
|--------------------------------------------------------|
| Hsu and Liu | Rao | MFPA |
| d | 5.4507 | 5.44 | 5.4512 |
| t | 0.292 | 0.293 | 0.29197 |
| \( g_1 \) | \(-3.4537 \times 10^{-5}\) | \(-1.4873 \times 10^{-3}\) | \(-2.3519 \times 10^{-5}\) |
| \( g_2 \) | \(1.3171 \times 10^{-4}\) | \(2.5802 \times 10^{-3}\) | \(-3.9619 \times 10^{-5}\) |
| \( F_{\text{min}} \) | 26.49912 | 26.50042 | 26.49995 |

In Table 4, MFPA has outperformed than CS in term of number of iteration and average elapsed time of each run. This is because MFPA can optimize the tubular column design problem almost seven times faster than CS in only 100 number of iteration. In this table, the method provided by Hsu and Liu as well as Rao are not feasible due to the violation of second constraint \(g_2\). Therefore, MFPA provides the best results among other algorithms in this particular problem.

7.2. Speed Reducer

A speed reducer [18] as illustrated in Figure 3 is a benchmark Mechanical Engineering design optimization problem with the face width (b), module of teeth (m), number of teeth on pinion (z), length of shaft 1 between bearings (l1), length of shaft 2 between bearings (l2), diameter of shaft 1 (d1), and diameter of shaft 2 (d2). Minimizing the total weight of the speed reducer is the objective of this
The objective function of speed reducer is given as following:

\[
\begin{align*}
\text{Minimize: } f(b, m, z, l_1, l_2, d_1, d_2) &= 0.7854bm^2(3.3333z^2 + 14.9334z - 43.0934) \\
&- 1.508b(d_1^2 + d_2^2) + 7.477(d_1^4 + d_2^4) \\
&+ 0.7854(l_1d_1^2 + l_2d_2^2)
\end{align*}
\]

Subject to:

\[
\begin{align*}
g_1 &= 27 \frac{bm^2z}{397.5} - 1 \ll 0 \\
g_2 &= 397.5 \frac{bm^2z^2}{1.93l_1^3} - 1 \ll 0 \\
g_3 &= \frac{mz d_1^4}{1.93l_1^3} - 1 \ll 0 \\
g_4 &= \frac{mz d_2^4}{1.93l_2^3} - 1 \ll 0 \\
g_5 &= \frac{1}{110d_1^3} \sqrt{\left(\frac{745l_1^4}{mz}\right)^2 + 16.9 \times 10^6} - 1 \ll 0 \\
g_6 &= \frac{1}{85d_2^3} \sqrt{\left(\frac{745l_2^4}{mz}\right)^2 + 157.5 \times 10^6} - 1 \ll 0 \\
g_7 &= \frac{mz}{40} - 1 \ll 0 \\
g_8 &= \frac{5m}{b} - 1 \ll 0 \\
g_9 &= \frac{12m}{b} - 1 \ll 0 \\
g_{10} &= \frac{1.5d_1 + 1.9}{l_1} - 1 \ll 0 \\
g_{11} &= \frac{1.1d_2 + 1.9}{l_2} - 1 \ll 0
\end{align*}
\]

The result provided by MFPA in optimizing speed reducer problem is shown in Table 5. The simple bounds of each design variable of the speed reducer problem are also shown in this table.
Table 5: Speed Reducer Problem Optimized by MFPA

| Bound | MFPA |
|-------|------|
| b     | 2.6-3.6 | 3.5 |
| m     | 0.7-0.8 | 0.7 |
| z     | 17-28   | 17  |
| l₁    | 7.3-8.3 | 7.3 |
| l₂    | 7.8-8.3 | 7.8005 |
| d₁    | 2.9-3.9 | 3.35021 |
| d₂    | 5.0-5.5 | 5.28668 |

Objective function value: 2996.2195
Popular size: 90
Maximum Number of iteration: 50
Time of each run (s): 1.325822

Table 6 shows a comparison of the results given by MFPA and other methods which are the result reported by Kuang, Rao, & Chen [19], Ray & Saini, [20], Akhtar, Tai, & Ray [21], Mezura-Montes, Coello, & Landa-Becerra [22] and Gandomi Yang, & Alavi [16]. Bold sets in Table 6 are violated sets which they do not fulfill the condition of constraint.

In Table 6, the result obtain by MFPA is better than those results reported by Akhtar, Tai, & Ray, Mezura-Montes, Coello, & Landa-Becerra and Gandomi Yang, & Alavi. Although the minimum objective value reported by Kuang, Rao, & Chen and Ray & Saini are better than those of MFPA, the reported minimum objective values are not feasible. This is due to the violation of sixth constraints ($g_6$) in the result of Kuang, Rao, & Chen. Meanwhile, for the result of Ray & Saini, there are violation in fifth and sixth constraints ($g_5$ & $g_6$). Therefore, MFPA has outperformed than other opponents as it has obtained the least objective value for speed reducer problem.

Table 6: Results Obtained from Different Algorithm on Optimizing Speed Reducer Problem

|          | Kuang, Rao, & Chen (1998) | Ray & Saini (2001) | Akhtar, Tai, & Ray (2002) | Mezura-Montes, Coello, & Landa-Becerra (2003) | Gandomi et al., (2013) | MFPA (2016) |
|----------|--------------------------|-------------------|--------------------------|---------------------------------------------|------------------------|-------------|
| $F_{min}$| 2876.117623              | 2732.9006         | 3008.08                  | 3025.005                                    | 3000.9810              | 2996.2195   |
| b        | 3.6                      | 3.514185          | 3.506123                 | 3.5015                                      | 3.5                    |
| m        | 0.7                      | 0.700005          | 0.700006                 | 0.7000                                      | 0.7                    |
| z        | 17                       | 17                | 17                       | 17                                          | 17                     |
| l₁       | 7.3                      | 7.497343          | 7.549126                 | 7.460181                                    | 7.6050                 |
| l₂       | 7.8                      | 7.8346            | 7.85933                  | 7.962143                                    | 7.8181                 |
| d₁       | 3.4                      | 2.9018            | 3.365576                 | 3.3629                                      | 3.3520                 |
| d₂       | 5                        | 5.0022            | 5.289773                 | 5.3090                                      | 5.2875                 |
| $g_1$    | -0.0996                  | -0.00777          | -0.0755                  | -0.0777                                     | -0.0743                |
| $g_2$    | -0.2203                  | -0.2012           | -0.1994                  | -0.2013                                     | -0.1983                |
| $g_3$    | -0.5279                  | -0.0360           | -0.4562                  | -0.4741                                     | -0.4349                |
| $g_4$    | -0.8769                  | -0.8754           | -0.8994                  | -0.8971                                     | -0.9008                |
| $g_5$    | -0.0433                  | 0.5395            | -0.0132                  | -0.0110                                     | -0.0011                |
| $g_6$    | **0.1821**               | **0.1805**        | -0.0017                  | -0.0125                                     | -0.0004                |
| $g_7$    | -0.7025                  | -0.7025           | -0.7025                  | -0.7022                                     | -0.7025                |
| $g_8$    | -0.0278                  | -0.0040           | -0.0017                  | -0.0006                                     | -0.0004                |
| $g_9$    | -0.5714                  | -0.5816           | -0.5826                  | -0.5831                                     | -0.5832                |
| $g_{10}$ | -0.0411                  | -0.1660           | -0.0796                  | -0.0691                                     | -0.0890                |
| $g_{11}$ | -0.0513                  | -0.0552           | -0.0179                  | -0.0279                                     | -0.0130                |
7.3. Gear Train
Sandgren has proposed a gear train mechanical design problem as shown in Figure 4 [23]. It is an unconstrained problem which has four integer variables. The objective of this problem is to produce a gear ratio as possible as 1/6.931. The objective function of this model is as follow:

\[
\text{Minimize: } f(T_d, T_b, T_a, T_f) = \frac{1}{6.931} - \frac{T_d T_b}{T_a T_f}
\]  (11)

Where \( T_i \) is the number of teeth of the gear \( i \) and they are all integers varying in the range of 12 to 60.

![Figure-4: Gear Train](image)

The result obtained by MFPA in optimizing gear train design problem is shown in Table 7. This table also shows the comparison result obtained by MFPA and CS [16]. In Table 8, it shows the comparison result obtained by MFPA and other methods which are the result reported by Sandgren [23], Kannan & Kramer [24] and Deb & Goyal [25].

|                  | CS         | MFPA      |
|------------------|------------|-----------|
| Population size  | 25         | 30        |
| Maximum Number of iteration | 5000   | 100       |
| \( T_d \)        | 19         | 17        |
| \( T_b \)        | 16         | 28        |
| \( T_a \)        | 43         | 60        |
| \( T_f \)        | 49         | 55        |
| Gear ratio       | 0.144281   | 0.144242  |
| \( F_{\text{min}} \) | \(-1.643 \times 10^{-6}\) | \(3.69 \times 10^{-5}\) |
| Time of each run | 3.25003    | 0.349064  |

In Table 7, MFPA has outperformed than CS in term of time of each run. This is because MFPA optimizes the gear train model almost ten times faster than CS although CS obtains better minimum objective value. In Table 8 shows that MFPA able to optimize gear train model and give a competitive solution as compare to other methods in a shorter time. Therefore, MFPA is a competitive algorithm in this gear train model as compare to other algorithms.
Table-8: Comparison Results of Different Optimization Algorithm in Optimizing Gear Train Model

| Algorithm            | Td (°) | Th (°) | Ta (°) | Tf (°) | Gear Ratio | F\text{min} |
|----------------------|--------|--------|--------|--------|------------|-------------|
| Sandgren (1990)      | 18     | 13     | 19     | 17     | 0.146667   | -2.387 × 10^{-3} |
| Kannan & Kramer (1994)| 22     | 15     | 16     | 28     | 0.144124   | 1.552 × 10^{-4}  |
| Deb & Goyal (1996)   | 45     | 33     | 49     | 60     | 0.144281   | -1.643 × 10^{-6} |
| MFPA (2016)          | 60     | 41     | 43     | 55     | 0.144242   | 3.69 × 10^{-5}   |

7.4. Tension/Compression Spring Design

Designing a spring that can fulfill a mechanical task is a crucial part in a design process. A few design variables are needed to be determined that fulfill the constraints of the problem. Therefore, optimization is here to tackle these problems. Tension or compression spring design is one of the mechanical problems as shown in Figure 5. Minimizing the weight of a tension or compression spring is the objective of this problem. In the same time, it has to fulfill the constraints of this problem which are minimum deflection, shear stress, surge frequency, as well as limits on outside diameter and on design variables.

There are three variables in this problem which are wire diameter \(x_1\), the mean coil diameter \(x_2\) and the number of active coils \(x_3\). The objective function of this problem is shown as follow:

\[
\text{Minimize: } f(\vec{x}) = (x_3 + 2)x_2 x_1^2
\]  

Subject to:

\[
g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \ll 0
\]

\[
g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3) - x_1^4} + \frac{1}{5108 x_1^2} - 1 \ll 0
\]

\[
g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \ll 0
\]

\[
g_4(\vec{x}) = \frac{x_2 + x_1}{1.5} - 1 \ll 0
\]

Where \(0.05 \ll x_1 \ll 2.0, 0.25 \ll x_2 \ll 1.3, \text{and } 2.0 \ll x_3 \ll 15.0\).
Figure-5: Tension/compression spring

Table 9 shows the best solution obtained from MFPA in optimizing tension or compression spring design problem. The best solution obtained from MFPA is then compared with the result obtained from other algorithms in Table 10.

Table 9: Best result obtained from MFPA

| Parameter                  | MFPA     |
|----------------------------|----------|
| Population size            | 90       |
| Maximum Number of iteration| 100      |
| $x_1$                      | 0.0517621|
| $x_2$                      | 0.358477 |
| $x_3$                      | 11.1866  |
| $F_{\text{min}}$           | 0.012665 |
| Time of each run (s)       | 1.619263 |

Table 10: Comparison results of different optimization algorithm in optimizing tension or compression spring design problem

| Parameter | SiC-PSO | G-QPSO(2) | T-Cell | MFPA |
|-----------|---------|----------|--------|------|
| $x_1$     | 0.051690| 0.051515 | 0.051622| 0.0517621|
| $x_2$     | 0.356750| 0.352529 | 0.355105| 0.358477 |
| $x_3$     | 11.287126| 11.538862| 11.384534| 11.1866 |
| $F_{\text{min}}$ | 0.012665| 0.012665| 0.012665| 0.012665|

The result at the second column in Table 10 is obtained from Simple Constrained Particle Swarm Optimizer (SiC-PSO) which is proposed by Cagnina and Susana at 2008 [26]. Leandro dos Santos Coelho has proposed Quantum-behaved Particle Swarm Optimizer (G-QPSO) [27] whereas Victoria S. Aragón and Susana C. Esquivel have proposed T-Cell algorithm [28]. Their results on optimizing tension or compression spring design problem are listed in this table at third and fourth column respectively. As can be seen in this table, all of the algorithms can achieve the minimum objective value of 0.012665. Therefore, MFPA is a competitive algorithm as other optimization algorithms in this problem.

7.5. Pressure Vessel

A compressed air tank is designed to withstand a working pressure of 3000 psi and has a minimum volume of 750 ft$^3$. It is made of a cylindrical pressure vessel capped at both ends by hemispherical heads as illustrated in Figure 6. ASME boiler and pressure vessel code are followed to design this compressed air tank. The objective of this problem is to minimize the total cost which including material cost, forming cost and welding cost. The design variables of this problem are thickness $x_1$, thickness of the head $x_2$, the inner radius $x_3$ and the length of the cylindrical section of the vessel $x_4$. The thickness $x_1$ and thickness of the head $x_2$ are discrete values which are integer multiples of 0.0625 inch [23].
The objective function of this model is as follows:

\[
\text{Minimize: } f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 \\
+ 3.1661x_1^2x_4 + 19.84x_1^2x_3
\]  

(13)

Subject to:

\[
g_1(\vec{x}) = -x_1 + 0.0193x_3 < 0 \\
g_2(\vec{x}) = -x_2 + 0.00954x_3 < 0 \\
g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3} \pi x_3^3 + 1296000 < 0 \\
g_4(\vec{x}) = x_4 - 240 < 0
\]

where \(1 \times 0.0625 < x_1, x_2 < 99 \times 0.0625\) and \(10.0 < x_3, x_4 < 200.0\).

Table 11 shows the best solution obtained from MFPA in optimizing pressure vessel problem. It also shows the comparison result obtained by MFPA and CS in this table [16]. The best solution obtained from MFPA is then compared with the result obtained from other algorithms such as Stochastic Ranking (SR) which is reported by Runarsson and Yao [29], Artificial Immune System Genetic Algorithm (AIS-GA) which is proposed by Bernardino, Barbosa and Lemonge [30] as well as T-cell algorithm which is proposed by Victoria S. Aragón and Susana C. Esquivel [28] in Table 12.

**Table-11:** Best result obtained from CS and MFPA in optimizing pressure vessel problem

|          | CS       | MFPA    |
|----------|----------|---------|
| Population size | 25       | 90      |
| Maximum Number of iteration | 15000    | 100     |
| \(x_1\)   | 0.8125   | 0.8125  |
| \(x_2\)   | 0.4375   | 0.4375  |
| \(x_3\)   | 42.098456| 42.09845|
| \(x_4\)   | 176.6365958| 176.6366 |
| \(F_{\text{min}}\) | 6059.7143348| 6059.7143 |
| Time of each run (s) | 2.50472 | 1.426891 |

**Table-12:** Comparison results of different optimization algorithm in optimizing pressure vessel problem

|          | SR          | AIS-GA      | T-Cell       | MFPA        |
|----------|-------------|-------------|--------------|-------------|
| \(x_1\)  | 1.1250      | 0.8125      | 0.8125       | 0.8125      |
| \(x_2\)  | 0.5625      | 0.4375      | 0.4375       | 0.4375      |
| \(x_3\)  | 58.1267     | 42.0931     | 42.098429    | 42.09845    |
In Table 11, MFPA has outperformed than CS in term of number of iteration and time of each run. This is because MFPA optimizes the pressure vessel problem almost one second faster than CS in only 100 number of iteration. In Table 12 shows that MFPA has produced outstanding result in optimizing pressure vessel problem as it gives the lowest minimum function value among other algorithms proposed by other researchers. Therefore, MFPA is a powerful algorithm in optimizing this pressure vessel problem.

8. Conclusion
In this study, MFPA is used to optimize five benchmarks Mechanical Engineering design problems. In tabular column design, MFPA has optimized the model almost seven times faster than CS and in the same time, enable to obtain the least magnitude of objective function. In speed reducer, MFPA has obtained the least magnitude of objective function compare to other algorithms proposed by other researchers in latest research. Furthermore, MFPA has optimized the gear train model ten times faster than CS. Apart from that, MFPA enable to optimize tension/compression spring design as other algorithms. Lastly, MFPA has optimized pressure vessel model one second faster than CS by only 100 number of iteration. Therefore, these results prove that MFPA is feasible in optimizing five benchmark Mechanical Engineering design problems as well as real world Mechanical Engineering design problem.

Acknowledgments
Financial support from the Malaysian government with cooperation of Universiti Tun Hussein Onn (UTHM) in the form of FRGS Vot 1490 is gratefully acknowledged.

References
[1] Gomez, A. G., & Gruender, J. L. (2011). Engineering Your Future: A Project-Based Introduction to Engineering. Wildwood: Great Lakes Press, Incorporated. Retrieved from https://books.google.com.my/books?id=kMOzNAEACAAJ
[2] Ravindran, A., Ragsdell, K. M., & Reklaitis, G. V. (2006). Engineering Optimization: Methods and Application. Journal of Chemical Information and Modeling (2nd ed., Vol. 53). Hoboken, New Jersey: John Wiley & Sons, Inc. http://doi.org/10.1017/CBO9781107415324.004
[3] Yang, X. S. (2010). Engineering Optimization: An Introduction with Metaheuristic Applications. Engineering Optimization: An Introduction with Metaheuristic Applications. Hoboken, New Jersey: John Wiley & Sons, Inc. http://doi.org/10.1002/9780470640425
[4] J. H. Holland, Adaptation in Natural and Artificial Systems. (University of Michigan Press, 1975).
[5] J. Kennedy and R. Eberhart, presented at the Proceedings of IEEE International Conference on Neural Networks, 1995 (unpublished).
[6] X. S. Yang, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (2012), Vol. 7445 LNCS, pp. 240-249.
[7] M. Sharawi, E. Emamy, I. A. Saroit and H. El-Mahdy, Int J Soft Comput Eng 4, 54-59 (2014).
[8] E. Emamy, H. M. Zawbaa, A. E. Hassanien, M. F. Tolba and V. Snášel, presented at the Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014, 2014 (unpublished).
[9] Y. Abdelaziz and E. S. Ali, Electr Pow Comp Syst 43, 1268-1277 (2015).
[10] Abdelaziz, E. Ali and S. A. Elazim, Int J Eng Sci Techno 19, 610-618 (2016).
[11] Glover, Understanding Flowers and Flowering: An Intergrated Approach (Oxford University Press, New York, 2007).
[12] Pavlyukevich, J Comput Phys 226, 1830-1844 (2007).
[13] A. M. Reynolds and M. A. Frye, Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search, e354 (2007).
[14] B. Alatas, Expert Syst Appl 37, 5682-5687 (2010).
[15] Rao, S. S. (2009). Engineering Optimization: Theory and Practice (4th ed.). Hoboken, New Jersey: John Wiley & Sons, Inc. http://doi.org/10.1080/03052150500066646
[16] Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. Engineering with Computers, 29(1), 17–35. http://doi.org/10.10107/s00366-011-0241-y
[17] Hsu, Y.-L., & Liu, T.-C. (2007). Developing a fuzzy proportional-derivative controller optimization engine for engineering design optimization problems. Engineering Optimization, 39(6), 679–700. http://doi.org/10.1080/03052150701252664
[18] Gandomi, A. H., & Yang, X.-S. (2011). Benchmark Problems in Structural Optimization. In S. Kooziel & X.-S. Yang (Eds.), Computational Optimization, Methods and Algorithms (pp. 259–281). Berlin, Heidelberg: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-20859-1_12
[19] KU, K. J., RAO, S. S., & CHEN, L. I. (1998). TAGUCHI-AIDED SEARCH METHOD FOR DESIGN OPTIMIZATION OF ENGINEERING SYSTEMS. Engineering Optimization, 30(1), 1–23. http://doi.org/10.1080/03052159808941235
[20] RAY, T., & SAINI, P. (2001). ENGINEERING DESIGN OPTIMIZATION USING A SWARM WITH AN INTELLIGENT INFORMATION SHARING AMONG INDIVIDUALS. Engineering Optimization, 33(6), 735–748. http://doi.org/10.1080/03052150108940941
[21] Akhtar, S., Tai, K., & Ray, T. (2002). A Socio-Behavioural Simulation Model for Engineering Design Optimization. Engineering Optimization, 34(4), 341–354. http://doi.org/10.1080/03052150212723
[22] Mezura-Montes, E., Coello, C. A. C., & Landa-Becerra, R. (2003). Engineering optimization using simple evolutionary algorithm. Proceedings 15th IEEE International Conference on Tools with Artificial Intelligence, (December 2003), 149–156. http://doi.org/10.1109/TAI.2003.1250183
[23] Sandgren, E. (1990). Nonlinear Integer and Discrete Programming in Mechanical Design Optimization. Journal of Mechanical Design, 112(2), 223–229. Retrieved from http://dx.doi.org/10.1115/1.2912596
[24] Kannan, B. K., & Kramer, S. N. (1994). An Augmented Lagrange Multiplier Based Method for Mixed Integer Discrete Continuous Optimization and Its Applications to Mechanical Design. Journal of Mechanical Design, 116(2), 405–411. Retrieved from http://dx.doi.org/10.1115/1.2919393
[25] Deb, K., & Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. Computer Science and Informatics, 26(1), 30–45. http://doi.org/citeulike-article-id:9625478
[26] Cagnina, L. C., Esquivel, S. C., Nacional, U., Luis, D. S., Luis, S., & Coello, C. A. C. (2008). Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer 1 Introduction 2 Literature review 3 Our proposed approach : SiC-PSO, 32, 319–326.
[27] Coelho, L. dos S. (2010). Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. Expert Systems with Applications, 37(2), 1676–1683. http://doi.org/10.1016/j.eswa.2009.06.044
[28] Aragón, V. S., Esquivel, S. C., & Coello, C. A. C. B. C. (2010). A modified version of a T-Cell Algorithm for constrained optimization problems. International Journal for Numerical Methods in Engineering, 84(3), 351–378. http://doi.org/10.1002/nme.2914

[29] Runarsson, T. P., & Yao, X. (2000). Stochastic ranking for constrained evolutionary optimization. IEEE Transactions on Evolutionary Computation. http://doi.org/10.1109/4235.873238

[30] Bernardino, H. S., Barbosa, H. J. C., & Lemonge, A. C. C. (2007). A hybrid genetic algorithm for constrained optimization problems in mechanical engineering. 2007 IEEE Congress on Evolutionary Computation. http://doi.org/10.1109/CEC.2007.4424532