Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

AN ELITE NON-DOMINATED SORTING MULTI-OBJECTIVE ARTIFICIAL BEE COLONY ALGORITHM FOR SUSTAINABLE & TECHNO-ECONOMIC MULTI-CRITERIA POWER FLOW OPTIMIZATION

Mr. Abhishek Bajirao Katkar Lecturer, Electrical Engineering, Government Polytechnic, Kolhapur, Maharashtra.
(Prof.) Dr H. T. Jadhav, Dean, Faculty of Science & Technology, SNDT Women's University, Mumbai, Maharashtra.

ABSTRACT
A metaheuristic strategy based on the Non-dominated Sorting-based Multi-objective Artificial Bee Colony (NS-MOABC) algorithm optimizes multi-criteria power flow in this work. This work uses the Artificial Bee Colony (ABC) algorithm, a non-dominated sorting approach, an external archive, adaptive learning, greedy selection, crowding distance, and elite search tactics. The elements maintain diversity and guide the quest with exceptional insights. A novel method is used in the bee phase to achieve convergence and diversity, achieving multi-objective optimization. ABC's core mechanics are preserved in the observer bee phase, which uses a modified fitness calculation to explore potential solutions. The scout bee phase always removes inefficient solutions and promotes variety through random solutions. To maintain an optimum and varied solution set in the swarm, non-dominated sorting and crowding distance approaches from the NSGA-II algorithm are used. A repository with a set capacity stores non-dominated solutions. The NS-MOABC algorithm is compared to state-of-the-art algorithms like NSGA-II and MOPSO. MOABC and the comparison NS-MOABC outperformed NSGA-II, MOPSO, and MOABC on six benchmark test issues. The 30-bus IEEE test system shows the algorithm's actual use, including cost, loss, and economic aspects for techno-economic sustainability. If compared to NSGA-II, MOPSO, and MOABC, the simulation results show that the suggested technique is effective and resilient. NS-MOABC is a simple, efficient, and resilient algorithm for MOOPF problems.

Keywords: Non-dominated Sorting-based Multi-objective Artificial Bee Colony (NS-MOABC),Non-dominated Sorting Genetic Algorithm(NSGA-II), Multi-objective Optimal Power Flow(MOOPF).

I. Introduction
Multi-objective optimisation problems (MOPs) are difficult because they have competing goals. [1] In Multi-Objective Problems (MOPs), there is no one solution that meets all objectives. The solver finds the Pareto-optimal set of trade-off solutions, where no alternative is inferior. Real-world Pareto-optimal sets might be enormous or infinite, growing exponentially with issue size. [3] Optimisation and decision-making are usually needed to solve multi-objective situations. System operators must balance numerous goals when operating and designing real-world power systems.[5] In power system control, optimum power flow (OPF) determines the best control variable modifications to minimize specified goal functions while meeting equipment and network restrictions. Numerous practical engineering and domain-specific challenges demand concurrent optimisation of numerous objective functions.[2] Multi-objective optimisation problems (MOPs) are difficult owing to conflicting objectives. A single solution that meets all objectives in Multi-Objective Problems (MOPs) is impracticable.[4] The solver finds the Pareto-optimal collection of trade-off solutions, where no alternative is inferior. Real-world issues' Pareto-optimal set may be large or infinite, expanding exponentially with size.[6]Real-world power system operators must weigh many goals when planning and operating. Power system management relies on optimum power flow (OPF) to find the best control variable changes to reduce objective functions while meeting equipment and network restrictions.
The literature presents many multi-objective artificial bee colony (MOABC) methods.[9] Despite extensive research on evolutionary and swarm intelligence algorithms like the non-dominated sorting

genetic algorithm II (NSGA-II), and multi-objective particle swarm optimisation (MOPSO), MOABC researchers still struggle to improve swarm diversity and address local convergence issues.[7] A MOABC approach by Akbari et al. incorporates a weight into the search equations throughout the employed bee and observer bee phases to manage the mutation relevance of the food supply. An adaptive grid-based system manages an external archive of search history, which bees use to adjust their search motions.[11] ABC solves real-world issues, however its sluggish convergence rate and premature convergence to local optima restrict it. [10] The employed and observer bee stages focus exploitation, whereas the scout bee phase emphasises exploration, but only one bee each generation. Similar to MOABC algorithms, restricted exploration may cause premature convergence to local optimum solutions.

No strategy is used to prevent premature convergence in the MOABC algorithm. Optimising approaches require a fitness assignment mechanism to guide the population towards non-dominated solutions. [12] Pareto-dominance is used to assess fitness at MOABC, limiting diversity. To maintain variety, MOABC enhances archive solutions with dominance. This diversity-achieving approach works only when the number of non-dominated possible solutions rises exponentially with issue size, according to Horoba et al. [14], MOOPF with three main objectives: total fuel cost, total emissions, and total real power loss. This paper applies the non-dominated sorting-based multi-objective artificial bee colony (NS-MOABC) algorithm. The ideal compromise solution is found via fuzzy logic. According to the 30-bus IEEE test system, the NS-MOABC model is efficient. Compared to three existing multi-objective algorithms NSGA-II, MOPSO, and MOABC—the simulation results show the proposed method's efficacy and resilience.[16] **Novelty and Goal are as below**

The artificial bee colony (ABC) algorithm is enhanced to provide a novel multi-objective optimisation method. Several crucial aspects are novel.:1. **Integration of Non-Dominated Sorting and Crowding Distance**: Enhancing the artificial bee colony (ABC) algorithm creates a novel multi-objective optimisation method. There are numerous major breakthroughs.[13]2 **Enhanced Employed Bee Phase**: A new bee phase method guides solutions towards convergence while maintaining variety. This fixes MOABC algorithms' premature convergence and sluggish advancement in high-dimensional situations.[15]3. **Use of an External Archive**: An external archive is utilized to reserve the elite population, storing search history using non-dominated sorting and crowding distance procedures. This helps in maintaining an optimal and diverse set of solutions.4. **Application to OPF Problems**: The research addresses the optimum power flow (OPF) problem as a multi-objective optimisation model that considers fuel cost, emissions, and real power loss[17].

This project aims to create NS-MOABC, a robust and efficient multi-objective optimisation algorithm that can tackle complicated optimisation problems with many competing objectives. The study seeks to:1. **Improve Diversity and Convergence**: Improve diversity maintenance and avoid premature convergence in MOABC algorithms by using new fitness assignment and bee phase tactics.2.**Validate the Effectiveness of NS-MOABC**: Compare the suggested method against multi-objective optimisers like NSGA-II, MOPSO, and MOABC to prove its usefulness.3.**Apply to Real-World Problems**: Use the 30-bus IEEE test system to demonstrate the NS-MOABC algorithm's practicality in power system operation and planning. By addressing these objectives, the research aims to contribute a powerful tool for solving multi-objective optimization problems, particularly in the context of optimal power flow, ensuring both techno-economic and environmental sustainability

Section 2: Multiobjective Optimal Power Flow Problem Formulation’ gives the multi-objective OPF problem's mathematical formulation. Section 3: The 'Non-dominated Sorting Multi-objective Artificial Bee Algorithm' describes the proposed NS-MOABC in detail. The 'Benchmark Test' section shows that NS-MOABC outperforms the other three MO optimizers on six typical benchmark functions. The study implements NS-MOABC to MOOPF, Simulation findings. Validation of NS-MOABC with help of statistical analysis. Finally, 'Conclusions' outlines the findings.

II. Multi-objective Optimal power Flow Problem Formulations

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

Real-world multi-objective optimisation problems (MOPs) include competing and incommensurable objectives, making a single optimum solution unattainable. [19] Instead, these challenges provide trade-offs between objectives, with no solution dominating another. These solutions are non-dominated, non-inferior, or Pareto-optimal. Pareto-optimal or non-dominated solutions exist in the decision space, while their counterparts in the objective space form the Pareto front (PF), also known as the real or global Pareto-front. [18] multi-objective optimisation has two objectives: Convergence: The obtained Pareto-front's closeness to the genuine one. Diversity: Pareto-front solution distribution. An effective optimiser should accomplish convergence and variety to give the Decision Maker (DM) a wide range of optimal alternatives. Understanding multi-objective optimisation terminology and concepts helps you understand the issue and solution areas.[20] Problem formulation for MOOPF problem finds the best control variables to minimise multiple objective functions under equality and inequality constraints. Formulate the MOOPF issue as:

$$\underset{x}{\text{Minimize}}\ f_i(x,u)\ i=1,\ldots,N_{obj}$$
$$\text{Subject to}: \begin{cases} g(x,u)=0 \\ h(x,u)\leqslant 0 \end{cases} \qquad (2)$$

where f_i is the i[th] objective function, N_{obj} is the number of objective functions, g is a set of constrain equations, and h is a set of formulated constrain in equations. x is the vector of dependent variables such as the slack bus power P_{G1}, the load bus voltage V_L, generator reactive power outputs Q_G, and the apparent power flow S_k. x can be expressed as:

$$x=[P_{G_1},V_{L_1},\ldots,V_{L_{P_Q}},Q_{G_1},\ldots,Q_{G_{N_G}},S_{L_1},\ldots,S_{L_{N_L}}] \qquad (3)$$

where N_{PQ}, N_G and N_L are the number of load buses, the number of generator buses, and the number of transmission lines, respectively. Here u is a set of the control variables such as the generator real power output P_G expect at the slack bus P_{G2}, the generator voltages V_G, the transformer tap setting T, and the reactive power generations of var source Q_C. Therefore, u can be expressed as:

$$u=[P_{G_2},\ldots,P_{G_{N_G}},V_{G_1},\ldots,V_{G_{N_G}},T_1,\ldots,T_{N_T},Q_{C_1},\ldots,Q_{C_{N_C}}] \qquad (4)$$

where N_T and N_C are the number of regulating transformers and the number of var compensators, respectively. Problem formulation for optimal power flow The MOOPF problem finds the best control variables to minimise multiple objective functions under equality and inequality constraints. Formulate the MOOPF issue Objective functions as mentioned here,

1.Minimization of fuel cost: Quadratic functions with sine components depict generating cost curves. These sine components show the steam admission valve openings' rippling effects [23]. Given valve loading effects, producing units' $/h fuel cost may be modelled as:

$$f_{\text{cost}}=\sum_{i=1}^{N_G}a_i+b_iP_{G_i}+c_iP_{G_i}^2+\left|d_i\times\sin\left(e_i\times\left(P_{G_i}^{\min}-P_{G_i}\right)\right)\right| \qquad (5)$$

where a_i, b_i, c_i, d_i, and e_i are the cost coefficients of the i[th] generator, and P_{Gi} is the real power output of thermal unit i.

2.Minimization of emission: The total ton/h emission of the atmospheric pollutants such as SOx and NOx caused by fossil-fueled thermal units can be expressed as [23]:

$$f_{emission}=\sum_{i-1}^{N_G}\alpha_i+\beta_iP_{G_i}+\gamma_iP_{G_i}^2 \qquad (6)$$

where α_i, β_i, and Υ_i are emission coefficients of the ith generating unit.

3.Minimization of transmission loss: The power flow solution gives all bus voltage magnitudes and angles. Then, the total MW active power loss in a transmission network can be described as follows:

$$f_{loss}=\sum_{k=1}^{N_L}g_k\left(V_i^2+V_j^2-2V_iV_j\cos(\delta_i-\delta_j)\right) \qquad (7)$$

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

where g_k is the conductance of k[th] branch, V_i, V_j, δ_i and δ_j are the voltage magnitudes and phase angles of terminal buses of branch k

4. Equality constraints: The equality constraints are the nonlinear power flow equations which are formulated as below:

$$0 = P_{G_i} - P_{D_i} - V_i \sum_{j=1}^{N_B} V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) \quad (8)$$

$$0 = Q_{G_i} - Q_{D_i} - V_i \sum_{i=1}^{N_B} V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) \quad (9)$$

where P_{Gi} is the injected active power at bus i, P_D is the demanded active power at bus i, Q_{Gi} is the injected reactive power at bus i, Q_{Di} is the demanded reactive power at bus i, G_{ij} is the transfer conductance between bus *i* and *j*, B_{ij} is the transfer susceptance between bus *i* and *j*, θ_{ij} is the voltage angle difference between bus i and j and N_B is the total number of buses.

5. Inequality constraints: These constraints are the set of continuous and discrete constraints that represent the system operational and security limits as follows:

$$\begin{aligned}
&P_{G_i}^{\min} \le P_{G_i} \le P_{G_i}^{\max} \quad i \in N_G \\
&Q_{G_i}^{\min} \le Q_{G_i} \le Q_{G_i}^{\max} \quad i \in N_G \\
&V_{G_i}^{\min} \le V_{G_i} \le V_{G_i}^{\max} \quad i \in N_G \\
&Q_{C_i}^{\min} \le Q_{C_i} \le Q_{C_i}^{\max} i \quad \in N_C \\
&T_i^{\min} \le T_i \le T_i^{\max} \quad i \in N_T \\
&V_{L_i}^{\min} \le V_{L_i} \le V_{L_i}^{\max} \quad i \in N_{PQ} \\
&S_{L_i} \le S_{L_i}^{\max} \quad i \in N_L
\end{aligned} \quad (10)$$

The penalty factor approach is used to handle inequality restrictions in this study. Each constraint-violating control vector will be fined by these factors. Thus, this control vector will be automatically erased next.

2.2 Multi-objective Artificial Bee Colony Algorithm

The MOABC takes honey bee foraging behaviour as inspiration. This colony has employed, onlooker, and scout bees. This algorithm optimises food sources. Each food supply solves the problem. The hired bees located these food sources for the dance area. Onlooker bees choose food by quality at the dance area. The MOABC has equal numbers of Employed and Onlooker bees. A Scout bee will randomly search the problem space for a new food source if it cannot be optimised in a few cycles. An external archive stores non-dominated solutions in the Pareto-based MOABC algorithm. [22] In a minimisation issue, two solutions are nondominated if neither has less value in all objectives than the other. We need an updating technique to add the latest archive solutions and assess if they dominate other solutions or archive members.

2.2.1. Initialization: To count food sources, a new variable called Food Number will be created during initialisation. The Food Number can be adjusted to half the population because each hired bee is a food source and the number of onlookers is equal. If a food supply cannot be optimised in several cycles, its employed bee will become a scout bee and then return to an employed bee after a random search. This is why scout bee populations have not been considered. This can be formulated as $L_{Bd} < xd < U_{Bd}$, in which d is the index of that parameter, x_d is the parameter and L_{Bd} and U_{Bd} are lower and upper bounds respectively. Each of these parameters can be considered as a dimension which will lead us to a search space. Hence, for a problem with *D* parameters or dimensions, a search space S through algorithm will associate a position vector $x = (x_1, x_2, \ldots, x_D)$ with each food source. Next, each food source will initialize through a function named *init(i, S)*, in which i is the index of the food source and

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

S is the search space that we have defined before. In this way, a D dimensional vector $x_i = (x_1, x_2, \ldots, x_D)$ will be assigned randomly to each food source i through the following equation: $\boldsymbol{x_{id} = L_{Bd} + rand[0, 1] * (U_{Bd} - L_{Bd})}$ **, ……(11)** where $d \in \{1, 2, \ldots, D\}$, $rand[0, 1]$ is a random number selected from a normal distribution in the range of zero and one, and L_{Bd} and U_{Bd} are lower and upper bounds along the d dimension respectively. Finally a $Trial_i$ variable will be assigned to each food source i in order to find food sources to be abandoned in the next iterations.

2.2.2 Send employed bees: To optimize food sources by their employed bees, the algorithm uses the archive since it contains the best solutions found so far. For this reason, each employed bee i would select an archive member randomly to calculate a temporary position called v_{id}. The position v_{id} is a copy of the position vector of the food source and one of its dimensions will be updated through the following equation: $\boldsymbol{v_{id} = x_{id} + w_1 rand[0, 1] (x_{id} - x_{kd})}$ ***…….(12)*** where i represents the food sources which is going to be optimized, k ∈ {1, 2, . . . , FoodNumber} and d ∈ {1, 2, . . . , D} are randomly chosen indexes. Although k is determined randomly, it has to be different from i. The random number rid is a real number chosen randomly in −1 to 1 span. It controls the production of neighbour food sources around x_{id} and represents the comparison of two food positions visually by a bee. The coefficient w_1 controls the importance of the food source k in the production of the new food source. As can be seen from Eq. (2), as the difference between the parameters of the x_{id} and x_{kd} decreases, the perturbation on the position x_{id} gets decreased, too. Thus, as the search approaches the optimum solution in the search space, the step length is adaptively reduced. After v_{id} has been updated, the value of objective functions will be calculated. If the result could dominate the related food sources results, then it will be replaced by the food source's vector. If not, the update was unsuccessful and Trial value of that food source will be incremented by one.

2.2.3 Send onlooker bees: The external archive will help all employed bees optimise their food source, and they will return to the hive to inform other bees. For this, the probability of each food source k advertised by the employed bee will be calculated.:

$$p_k = \frac{fit\left(\vec{x}_k\right)}{\sum_{m=1}^{FoodNumber} fit\left(\vec{x}_m\right)}, \qquad (13)$$

$fit\left(\vec{x}_m\right)$ is the probability of the food source proposed by the employed bee k which is proportional to the quality of food source. In the MOABC, the quality of a food source depends on the number of food sources dominated by the food source k. This can be formulated using the following equation:

$$fit\left(\vec{x}_m\right) = \frac{dom\,(m)}{FoodNumber}, \qquad (14)$$

where *dom(m)* is the function that returns the number of food sources dominated by food source m. After that, onlookers can use a roulette wheel to choose a food source advertised by the employed bee k based on its probability. The onlooker bee updates its position using the following equation if the newly discovered food source dominates the old one: $\boldsymbol{v_{id} = x_{id} + w_2 rand[0, 1] (x_{id} - x_{kd})}$ *……..(15)* where coefficient w_2 controls the importance of information provided by an employed bee k. This probabilistic approach optimises food sources and makes Archiving them more likely. Thus, each food source should be assessed to determine its quality so the viewer can choose.

2.2.4 Send scout bees: The algorithm will find abandoned food sources to replace them. Each worker or observer bee optimising a food source has two outcomes. First, she cannot finish it, increasing Trial of that food source by one. Second, it can reset Trial value to zero. Thus, if the Trial reaches *Max_Trial*, its employed bee will become a scout bee and reset it to zero after a random search. The *Max_Trial* parameter is set manually at the value of 60. The scout randomly moves and replaces the abandoned food source if it has better nectar. Assume that the abandoned source is x_i and j ∈ {1, 2, . . . , D}, then the scout discovers a new food source to be replaced with x_i. This operation can be defined as follows: $\boldsymbol{v_{id} = L_{Bd} + rand[0, 1] * (U_{Bd} - L_{Bd})}$ **.** …(16) After each candidate source position v_{ij} is produced and

then evaluated by the artificial bee, its performance is compared with that of its old one. This deficiency primarily arises because the solution search equation excels in exploration but lacks in exploitation, so prior necessity to work in such domain is for multi-objective optimization.

2.3 Non-dominated Sorting Genetic Algorithm II (NSGA-II):

The Non-dominated Sorting Genetic Algorithm II (NSGA-II), developed by Deb et al. in 2002, is a pioneering multi-objective optimisation algorithm recognised for its effectiveness in producing a varied array of high-quality solutions. This algorithm incorporates multiple innovative mechanisms, such as non-dominated sorting, crowding distance, and elitism, to effectively approximate the Pareto front. Elitism maintains superior solutions through generations, thereby augmenting the algorithm's resilience. Despite its extensive utilisation, NSGA-II faces challenges including computational complexity in large-scale problems and potential diversity loss in complex landscapes. These constraints have prompted continuous investigation to modify non-dominated sorting and other methodologies to tackle the intricacies of multi-objective optimisation. The fast nondominated sorting approach identifies two key criteria for solutions in a population: the dominated count n_i which are number of solutions and S_i which are the set of solutions dominated by solution i. Initially, the first nondominated front is formed by all solutions with n_i=0. For each solution j in S_i, its dominated count is decreased by one. If j's count reaches zero, it is added to a separate list j (second nondominated front). This process is repeated for each member of j to identify subsequent fronts until all solutions are sorted into different nondominated fronts.

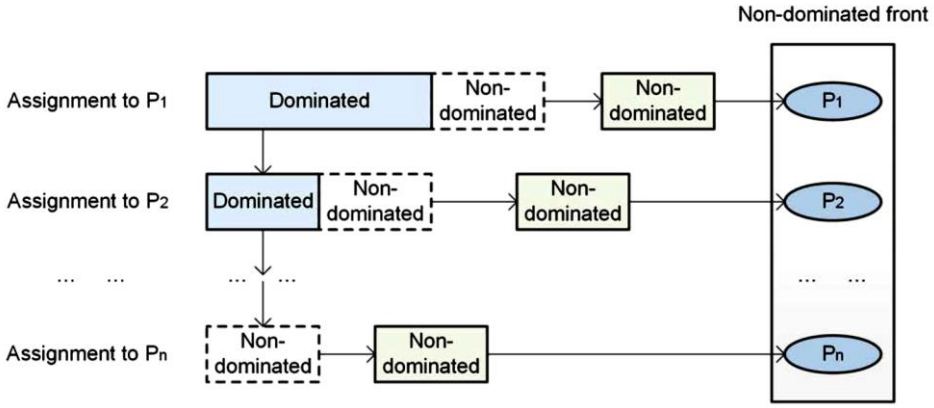


Figure 1: Process flow of Non-dominated sorting procedure for MOO

3. Non-dominated Sorting based Multi-objective Artificial Bee Colony Algorithm:

The Artificial Bee Colony (ABC) algorithm is effective for single-objective optimisation; however, it encounters challenges in multi-objective optimisation, particularly in acquiring and sustaining a diverse array of Pareto-optimal solutions. The non-dominated sorting-based multi-objective artificial bee colony (NS-MOABC) algorithm is proposed to address this issue.[1] This extension of the ABC algorithm addresses multi-objective optimisation problems by guiding solutions towards convergence while preserving diversity through an innovative method in the employed bee phase. The NS-MOABC algorithm utilises a fitness assignment strategy based on Pareto ranking and crowding distance, drawing from NSGA-II,[14] to direct the population towards the optimal Pareto front while maintaining both superior and diverse solutions. The algorithm consists of five distinct phases: population initialisation, employed bee phase, onlooker bee phase, scout bee phase, and archive update. Furthermore, two modifications improve exploitation while maintaining exploration: altered solution search equations in the employed and onlooker bee phases, and a revised neighbourhood search operator. The crowding distance operator enhances the process by assessing the density of solutions surrounding each point, [18] thereby promoting a balanced and efficient optimisation procedure. The integration of these features renders the NS-MOABC algorithm an effective solution for multi-objective optimisation. below flowchart of NS-MOABC Algorithm.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

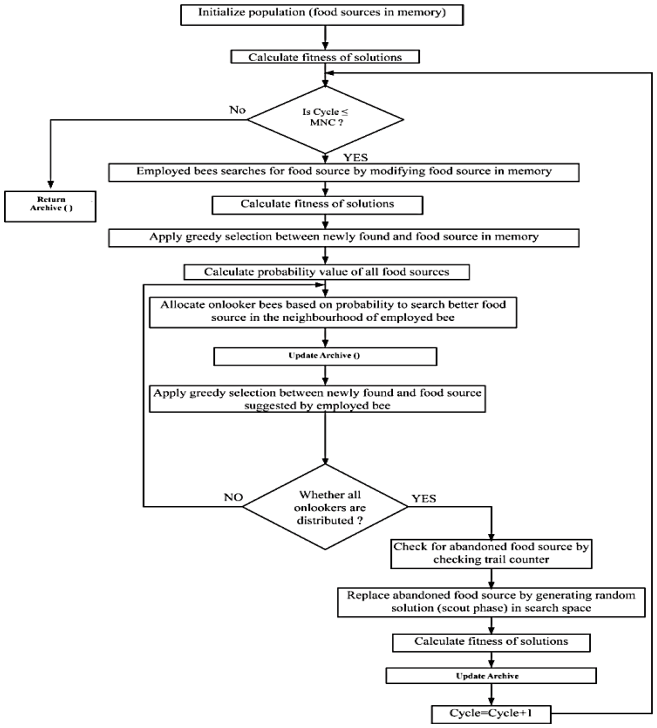


Figure 2: Flow chart of NS-MOABC Algorithm

4. Experimental Studies on Benchmark Test Functions: The NSABC algorithm is compared to 3 state-of-the-art MO Algorithms and variants of multi-objective artificial bee colony algorithms: S-MOABC/NS [46], MOABC [45], and 6 benchmark problems of varying nature and complexity [18]. To fully evaluate the performance of the NS-MOABC algorithm without a biased conclusion towards some chosen problems, we employed four 2-objective and two 3-objective benchmark functions. The formulas of these functions are presented below Table 1, Here multi-objective test problems used to evaluate the NS-MOABC algorithm. Select test problems include ZDT1 to ZDT6 for 2 objective optimal functions and DTLZ2 and DTLZ6 for 3 or more objective optimal solutions to evaluate efficacy on complex test problems.

Performance measures: In order to facilitate the quantitative assessment of the performance of a multi-objective optimization algorithm, two performance metrics are taken into consideration: (1) convergence metric (γ) (2) diversity metric (Δ) [20].

4.1 Convergence metric: This metric measures the extent of convergence to POS

$$\Upsilon = \frac{\sum_{i=1}^{N} d_i}{N} \qquad (20)$$

where N is the number of nondominated solutions obtained with an algorithm and d_i is the Euclidean distance between each of the nondominated solutions and the nearest member of the true Pareto optimal front. The average of these distances is used as the convergence metric γ.

4.2 Diversity metric: This metric measures the extent of spread achieved among the obtained solutions. Here, a set of solutions that spans the entire Pareto optimal region, it defined as:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}} \qquad (21)$$

where d_i is the Euclidean distance between consecutive solutions in the obtained nondominated set of solutions and N is the number of nondominated solutions obtained by an algorithm. d is the average value of these distances. d_f and d_l are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained nondominated set.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

SN	Benchmark Function	Objective Function	Pareto-optimal set	Corresponding Pareto-optimal region	Variable (n)	range
1.	ZDT1	Minimize $f_1(x) = x_1$ Minimize $f_2(x) = g(x)\left[1 - \sqrt{x_1/g(x)}\right]$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right) \Big/ (n-1)$	Convex,	$0 \le x_i^* \le 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 30$.	30-variables (n = 30)	[0, 1]
2.	ZDT2	Minimize $f_1(x) = x_1$ Minimize $f_2(x) = g(x)[1 - (x_1/g(x))^2]$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right) \Big/ (n-1)$	Non-convex	$0 \le x_i^* \le 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 30$.	30-variables (n = 30)	[0, 1]
3.	ZDT3	Minimize $f_1(x) = x_1$ Minimize $f_2(x) = g(x)\left[1 - \sqrt{x_1/g(x)} - \frac{x_1}{g(x)}\sin(10\pi x_1)\right]$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right) \Big/ (n-1)$	Dis-continuous	$0 \le x_i^* \le 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 30$. (not all points lie on the Pareto optimal front)	30-variables (n = 30)	[0, 1]
4.	ZDT6	Minimize $f_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ Minimize $f_2(x) = g(x)[1 - (f_1(x)/g(x))^2]$ $g(x) = 1 + 9\left[\left(\sum_{i=2}^{n} x_i\right) \Big/ (n-1)\right]^{0.25}$	Non-convex	$0 \le x_i^* \le 1$ and $x_i^* = 0$ for $i = 2, 3, \ldots, 10$.	10-variables (n = 10)	[0, 1]
5.	DTLZ2	Minimize $f_1(x) = (1 + g(x_M))\cos(x_1\pi/2)\cdots\cos(x_{M-1}\pi/2)$ Minimize $f_2(x) = (1 + g(x_M))\cos(x_1\pi/2)\cdots\sin(x_{M-1}\pi/2)$ $\vdots \quad \vdots$ Minimize $f_M(x) = (1 + g(x_M))\sin(x_1\pi/2)$ Subject to $0 \le x_i \le 1$, for $i = 1, \ldots, n$ Where $g(x_M) = \sum_{x_i \in x_M} (x_i - 0.5)^2$	Spherical	$x_i^* = 0.5$, $(x_i^* \in x_M)$ Objective functions satisfy $\sum_{m=1}^{M} (f_m^*)^2 = 1$.	n = M + k _ 1.	use $k = \lvert x_M \rvert = 10$.
6.	DTLZ6	Minimize $f_1(x) = x_1$ Minimize $f_2(x) = x_2$ $\vdots \quad \vdots$ Minimize $f_{M-1}(x) = x_{M-1}$ Minimize $f_M(x) = (1 + g(x_M))h(f_1, f_2, \ldots, f_{M-1}, g)$ Subject to $0 \le x_i \le 1$, for $i = 1, \ldots, n$ Where $g(x_M) = 1 + \frac{g}{\lvert x_M \rvert}\sum_{x_i \in x_M} x_i$ $h = M - \sum_{i=1}^{M-1}\left[\frac{f_i}{1+g}(1 + \sin(3\pi f_i))\right]$	2^{M-1} Disconnected	functional g requires k = |xM| decision variables	n = M + k _ 1.	use $k = \lvert x_M \rvert = 10$.

Table 1: Benchmark test functions of ZDT and DTLZ suite

4.3 Computational complexity of NS-MOABC: Only a minor modification to the ABC algorithm's search equation in the bee phase makes the Non-dominated Sorting Artificial Bee Colony (NSABC) algorithm easy to implement and rationally computationally complex. The complexity, defined by function value comparisons per iteration, includes updating the archive using non-dominated sorting and searching in the employed and onlooker bee phases. The complexities include $O(M\times(NP)^2)$ for initial archive updates, $O(M\times(2NP))$ for bee phases, and $O(M\times(S+NP)^2) + O(M\times(S+NP))\log(S+NP)$ for iterative updates using non-dominated sorting and crowding distance estimation. Thus, the complexity is quadratic, matching modern multi-objective optimisation algorithms. M is the number of objectives, NP the swarm size, and S the archive size.

4.4. Simulation Parameters/Experimental setting

All experiments are run on MATLAB 2022(b) in Windows on a 64-bit 3.40 GHz Intel(R) Core(TM) i5-3770 PC with 4GB RAM. Set swarm size to 100. Three successful nature-inspired multi-objective optimisation algorithms—NSGA-II [18], MOPSO [19], and MOABC [12]—were used to evaluate the proposed NS-MOABC algorithm. This section uses the termination criterion compare algorithms with a fair time measure in all experiments. **NS-MOABC settings:** NS-MOABC parameter set[1] as swarm size is 100 and scout bees are limited to one [4]. The limit is NP×D/2 [6], with decision variables having a dimension of 30. Simulations are run 30 times for each test problem. The algorithm terminates after 300,000 function evaluations. T is 100,000 [15], and bi-objective problems have 100 archive size and three-objective problems 150. For the MOABC proposed in [4], a colony size of 50, archive size A = 100 was adopted. For **NSGA-II settings:** The original NSGA-II algorithm uses Simulated Binary Crossover (SBX) and Polynomial mutation. We use a population size of 100. Crossover probability *pc* = 0.9 and mutation probability is *pm* = 1/ n, where n is the number of decision variables. The distribution indices for crossover and mutation operators as η_c μ = 20 and η_m μm = 20 respectively. **MOPSO settings:** MOPSO used a population of 100 particles, a repository size of 100 particles, a mutation rate of 0.5, and 30 divisions for the adaptive grid. The detailed implementation and parameters setting for this MOPSO version can be refer to [19].

4.5 Simulations Results of Benchmark function Test:

Three algorithms' optimal fronts for each two objective functions are shown in Figs. 10–13. Continuous lines represent the Pareto optimal front, while mark spots represent algorithm-found nondominated solutions. When given 10,000 function evaluations for four algorithms on ZDT1

function, Table 3 shows that MOPSO performs one order of magnitude better in convergence metric than NSGA-II and MOABC but worse than NS-MOABC. Diversity metric shows that all algorithms perform similarly on ZDT1. Fig. 10 shows that NS-MOABC, MOPSO, and MOABC can find a diverse and well-distributed solution set. However, NSGA-II only finds a sparse distribution and cannot archive ZDT1's true Pareto front.

ZDT1	NS-MOABC	MOABC	NSGA-II	MOPSO
Convergence Metric				
Average	8.5564e-004	2.3249e-002	2.2378e-001	1.544e-003
Median	8.3555e-004	2.2355e-002	1.4510e-001	1.4522e-003
Best	7.666e-004	1.8967e-002	7.2172e-002	8.4812e-004
Worst	1.1510e-003	2.7409e-002	8.7448e-001	2.1314e-003
Std	8.0345e-005	2.6157e-003	2.3816e-001	6.6847e-004
Diversity Metric				
Average	6.1502e-001	3.3161e-001	5.9462e-001	6.9875e-001
Median	6.0738e-001	3.3118e-001	5.2794e-001	6.8883e-001
Best	5.3456e-001	2.8798e-001	4.5820e-001	6.7825e-001
Worst	6.7678e-001	3.7436e-001	9.1435e-001	7.2612e-001
Std	4.4416e-002	2.4224e-002	1.5659e-001	2.5562e-002

Table 2 Comparison of performance on ZDT1.

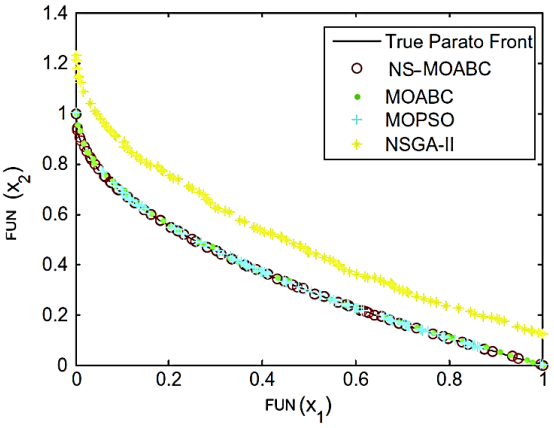


Figure 3: Pareto fronts obtained by NS-MABC, MOPSO, MOABC, and NSGA-II on ZDT1.

The ZDT2 performance measures show that NS-MOABC and MOABC have better convergence and diversity than MOPSO and NSGA-II. Table 4 shows that after 10,000 function evaluations, NS-MOABC and MOABC outperform MOPSO and NSGA-II in convergence and diversity metrics by three orders of magnitude. Table 4 shows that NS-MOABC outperforms MOABC in diversity metric by an order of magnitude. Fig. 11 shows that MOPSO and NSGA-II fail this test function and cannot reach the true Pareto front. On ZDT3 and ZDT6, the algorithms perform similarly to ZDT1 and ZDT2.

ZDT2	NS-MOABC	MOABC	NSGA-II	MOPSO
Convergence Metric				
Average	8.3512e-004	1.0123e-003	2.9576e-001	1.6120e-001
Median	8.3872e-004	9.5637e-004	1.8205e-001	8.5831e-002
Best	6.2366e-004	7.2238e-004	1.0676e-001	6.5377e-004
Worst	4.7323e-003	1.5790e-003	9.8318e-001	6.2214e-001
Std	1.4889e-001	2.5480e-004	2.6388e-001	6.6571e-004
Diversity Metric				
Average	6.7709e-002	2.9452e-001	7.6312e-001	6.3702e-001
Median	6.3879e-002	2.9285e-001	7.2899e-001	6.3710e-001
Best	6.1218e-002	2.5130e-001	4.6627e-001	5.2815e-001
Worst	7.5828e-002	3.3121e-001	1.051e+0001	7.2202e-001
Std	1.4641e-002	2.2247e-002	2.3667e-001	7.7398e-002

Table 3: Comparison of performance on ZDT2.

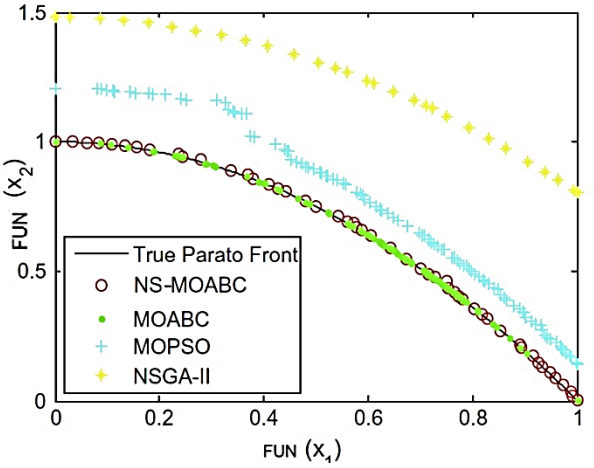
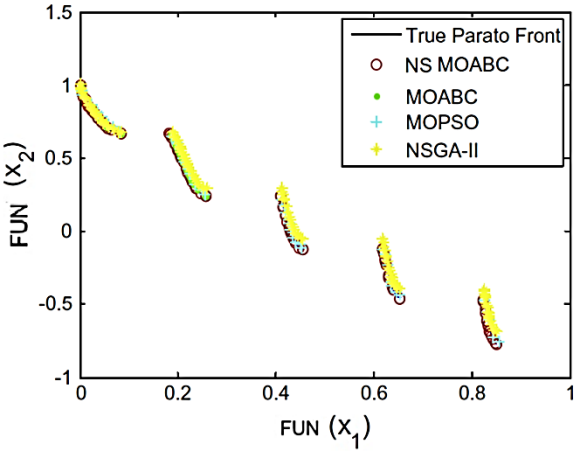


Figure 4: Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II on ZDT2.

ZDT3	NS-MOABC	MOABC	NSGA-II	MOPSO
Convergence Metric				
Average	4.0879e-003	1.5651e-003	3.3742e+000	1.4421e-002
Median	4.0650e-003	1.5286e-003	3.9788e+000	3.5313e-003
Best	3.5441e-003	1.2921e-003	6.9286e-001	2.6442e-003
Worst	4.7340e-003	2.1316e-003	4.7279e+000	5.8541e-002
Std	2.3874e-004	2.2484e-004	1.4832e+000	2.4754e-002
Diversity Metric				
Average	6.3688e-002	6.6157e-001	1.1239e+000	6.961e-001
Median	6.4590e-002	6.6425e-001	1.1287e+000	7.0544e-001
Best	5.5776e-002	6.3294e-001	1.0431e+000	5.0895e-001
Worst	6.7271e-002	6.8455e-001	1.2584e+000	8.7471e-001
Std	1.3804e-002	1.8562e-002	8.8924e-002	1.4362e-001

Table 4: Comparison of performance on ZDT3.



Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

Figure 5: Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II on ZDT3.

ZDT6	NS-MOABC	MOABC	NSGA-II	MOPSO
Convergence Metric				
Average	5.0234e-004	3.9788e-003	3.3542e+000	5.152e-001
Median	1.0799e-004	2.8135e-005	3.9788e+000	8.2562e-004
Best	9.6435e-005	1.9742e-005	6.9286e-001	6.5926e-004
Worst	1.7575e-003	2.6706e-002	4.7279e+000	2.5340
Std	7.4309e-001	8.9640e-003	1.4832e+000	1.1370
Diversity Metric				
Average	7.9531e-002	6.0221e-001	1.1229e+000	6.6278e-001
Median	7.6852e-002	4.7418e-001	1.1287e+000	6.741e-001
Best	6.0819e-002	4.3842e-001	1.0223e+000	5.964e-001
Worst	1.1895e-001	1.2217e+000	1.2859e+000	7.164e-001
Std	1.9428e-002	2.7898e-001	8.8814e-002	4.65e-002

Table 6 Comparison of performance on ZDT6.

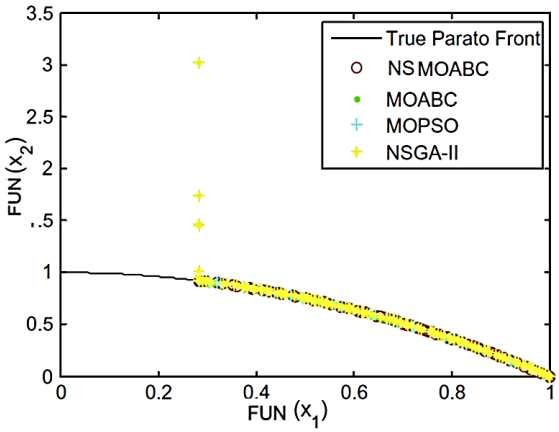


Figure 6: Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II on ZDT6.

Three objective functions: Optimisation results for three objective DTLZ series problems from NS-MOABC, MOPSO, MOABC, and NSGA-II algorithms are shown in Tables 7 and 8. Figs. 14–17 show the true Pareto optimal front and four algorithms' optimal fronts for DTLZ2 and DTLZ6.

DTLZ2	NS-MOABC	MOABC	NSGA-II	MOPSO
Convergence Metric				
Average	2.8471e-003	7.6836e-003	8.7363e-002	3.89231e-003
Median	2.8682e-003	6.7337e-003	8.9419e-002	3.81261e-003
Best	2.5637e-003	4.9233e-003	6.0319e-002	3.71302e-003
Worst	3.1186e-003	1.2708e-002	1.1310e-001	3.9421e-003
Std	1.5427e-004	2.4500e-003	1.6418e-002	8.1354e-005
Diversity Metric				
Average	4.3211e-002	4.1177e-001	6.0903e-001	4.1162e-001
Median	4.2736e-002	4.1107e-001	5.0881e-001	4.0565e-001
Best	3.9512e-002	3.5942e-001	3.7879e-001	3.9533e-001
Worst	4.6534e-002	4.7267e-001	7.3358e-001	4.4691e-001
Std	2.2815e-003	3.1091e-002	1.8498e-002	2.0232e-002

Table 5: Comparison of performance on DTLZ2.

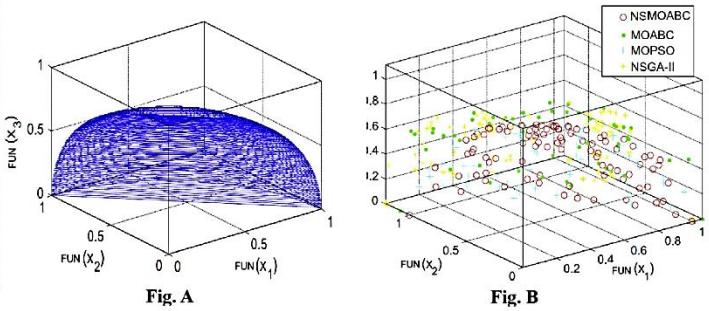


Fig. A **Fig. B**

Figure 7: (A) The true Pareto-optimal front on DTLZ2 , (B)Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II on DTLZ2.

Table 7 shows that all convergence metric algorithms performed well on DTLZ2 function. However, NSGA-II performs one order of magnitude worse in convergence metric than the other three algorithms, while NS-MOABC performs one order of magnitude better in diversity metric. Figure 15 shows that the NS-MOABC front is uniformly distributed on DTLZ2's true Pareto optimal front. Search space for DTLZ6 has 2M - 1 disconnected Pareto-optimal regions. An algorithm must maintain subpopulations in different Pareto-optimal regions in this problem. Table 8 shows that NS-MOABC outperforms all other algorithms in convergence and diversity. Fig 17 shows that NS-MOABC finds a diverse and well-distributed solution set for this problem. The NSGA-II cannot store DTLZ6's true Pareto front. shows that the proposed NS-MOABC algorithm generates better spread solutions but poor convergence. This shows that the proposed method can predict a few solutions on the two disconnected Pareto fronts.

DTLZ6	NS-MOABC	MOABC	NSGA-II	MOPSO
Convergence Metric				
Average	1.5598e-002	2.9901e-002	6.6160e-001	1.6426e-002
Median	1.5133e-002	2.7952e-002	2.8640e-001	1.7221e-002
Best	1.2331e-002	1.9106e-002	1.6724e-001	8.3231e-003
Worst	2.0474e-002	3.7997e-002	3.1497e+000	2.9165e-002
Std	2.6337e-003	6.7314e-003	9.2366e-001	8.3412e-003
Diversity Metric				
Average	5.2103e-002	5.4715e-001	5.7305e-001	5.4362e-001
Median	5.1993e-002	5.5144e-001	5.5839e-001	5.4445e-001
Best	4.6657e-002	5.0172e-001	4.5813e-001	5.184e-001
Worst	5.9160e-002	6.0581e-001	7.0776e-001	5.6638e-001
Std	3.7964e-003	2.9597e-002	7.2624e-002	1.8843e-002

Table 6: Comparison of performance on DTLZ6.

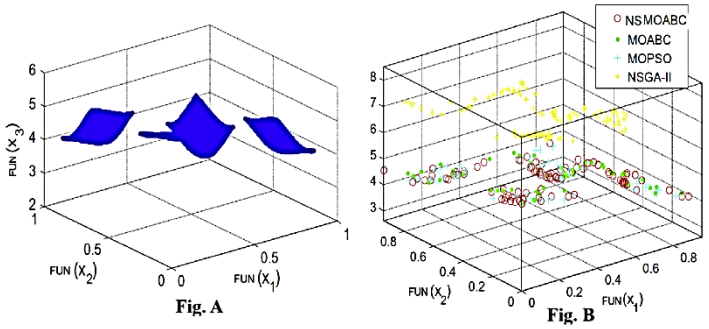


Fig. A **Fig. B**

Figure 8: (A)The true Pareto-optimal front on DTLZ6, (B)Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II on DTLZ6.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

5. MOOPF case study on IEEE 30 Bus system

This section details the proposed multi-objective OPF solution. NS-MOABC implementation for multi-criteria power flow optimisation. The following steps should be repeated to solve the OPF problem with multi-objective algorithms.

Step 1: Input the parameters of power system, parameters of the NS-MOABC algorithm, and lower and upper limits of each variable.

Step 2: Transfer the constraint multi-objective problem to an unconstraint one as follows:

$$F(x)=\begin{bmatrix}F_1(x)\\F_2(x)\\F_3(x)\end{bmatrix}=\begin{bmatrix}f_{cost}(x)+Z_1\left(\sum_{j=1}^{N_{eq}}(g_i(x))^2\right)+Z_2\left(\sum_{j=1}^{N_{ueq}}(\mathrm{Max}[0,-h_i(x)])^2\right)\\ f_{emission}(x)+Z_1\left(\sum_{j=1}^{N_{eq}}(g_i(x))^2\right)+Z_2\left(\sum_{j=1}^{N_{ueq}}(\mathrm{Max}[0,-h_i(x)])^2\right)\\ f_{loss}(x)+Z_1\left(\sum_{j=1}^{N_{eq}}(g_i(x))^2\right)+Z_2\left(\sum_{j=1}^{N_{ueq}}(\mathrm{Max}[0,-h_i(x)])^2\right)\end{bmatrix} \quad (22)$$

where N_{eq} and N_{ueq} are the number of equality and inequality constraints, respectively. $g_j(x)$ and $h_j(x)$ are the equality and inequality constraints, respectively, and Z_1 and Z_2 are the penalty factors.

Step 3: Produce the initial NS-MOABC population. N × M(N≥ 2, M≥2) individuals based on state variables should be randomly generated

Step 4: Calculate the objective functions value for each bee in each hive, sort them based on nondomination, and store nondominated solutions in the external archive EA of each hive.

Step 5: Update the position of each bee in each hive according to comprehensive learning mechanism. If any element of each bee breaks its limit then its position is fixed to the maximum minimum operating point.

Step 6: Update each EA of each hive according to greedy selecting strategy, sort the EA based on nondomination, and select the solutions to stay in EA. If the number of nondominated solutions exceeds the allocated the size of EA, apply crowding distance to remove the crowded members.

Step 7: If the current iteration number obtains the preordained maximum iteration number, the algorithm is stopped, otherwise go to step 4.

Sato et al. [71] propose a novel idea that increasing selection pressure towards the Pareto-front scales a multi-objective optimisation algorithm for more objectives. Our NS-MOABC uses this idea. We modify the employed bee's search equation and introduce a Pareto-dominant augmented population. This method reduces dominated solutions in successive iterations, increasing selection pressure. Thus, the NS-MOABC outperforms other three-objective optimisation algorithms.

Pseudocode for NSABC-based MOOPF Algorithm

Step 1: Input the parameters of the power system, parameters of the NSABC algorithm, and the lower and upper limits of each variable

```
def initialize_parameters():
    # Define power system parameters, algorithm parameters, and variable limits
    parameters = {
        'power_system': {...},
        'NSABC_parameters': {...},
        'variable_limits': {'lower': [...], 'upper': [...]}
    }
    return parameters
```

Step 2: Transfer the constrained multi-objective problem to an unconstrained one using penalty factors Z1 and Z2

```
def transfer_to_unconstrained(parameters):
    # Define objective functions with penalty factors Z1 and Z2
    def F(x):
        return [
            fuel_cost(x) + Z1 * sum([g_j(x) for g_j in constraints_eq]),
            emission(x) + Z2 * sum([h_j(x) for h_j in constraints_ineq]),
            power_loss(x) + Z2 * sum([h_j(x) for h_j in constraints_ineq])
        ]
    return F
```

Step 3: Produce the initial NSABC population based on state variables

```
def produce_initial_population(parameters):
    N = parameters['NSABC_parameters']['N']  # Number of initialized hives
    M = parameters['NSABC_parameters']['M']  # Control variables
    D = parameters['NSABC_parameters']['D']  # Bees in each hive

    population = [[[random_state_variable() for _ in range(D)] for _ in range(M)] for _ in range(N)]
    return population
```

Step 4: Calculate the objective function values for each bee, sort based on non-domination, and store non-dominated solutions in the external archive of each hive

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

```
def evaluate_population(population, F):
    evaluated_population = [[F(bee) for bee in hive] for hive in population]
    sorted_population = non_dominated_sorting(evaluated_population)
    archive = store_non_dominated(sorted_population)
    return evaluated_population, sorted_population, archive
```

Step 5: Update the position of each bee in each hive according to a comprehensive learning mechanism

```
def update_positions(population, evaluated_population):
    updated_population = []
    for hive in population:
        updated_hive = []
        for bee in hive:
            if bee_exceeds_limits(bee):
                updated_bee = fix_to_max_min(bee)
            else:
                updated_bee = learn_from_best(bee, evaluated_population)
            updated_hive.append(updated_bee)
        updated_population.append(updated_hive)
    return updated_population
```

Step 6: Update each external archive (EA) according to a greedy selecting strategy, sort based on non-domination, and apply crowding distance

```
def update_external_archive(archive, evaluated_population):
    for i, hive_archive in enumerate(archive):
        sorted_hive_archive = non_dominated_sorting(hive_archive + evaluated_population[i])
        archive[i] = select_non_dominated(sorted_hive_archive)
        if len(archive[i]) > max_size:
            archive[i] = apply_crowding_distance(archive[i])
    return archive
```

Step 7: Repeat steps 4-6 until the maximum iteration number is reached

```
def NSABC_MOOPF(parameters):
    F = transfer_to_unconstrained(parameters)
    population = produce_initial_population(parameters)
    archive = None
    for iteration in range(parameters['NSABC_parameters']['max_iterations']):
        evaluated_population, sorted_population, archive = evaluate_population(population, F)
        population = update_positions(population, evaluated_population)
        archive = update_external_archive(archive, evaluated_population)

    return archive
```

5.1 Best compromise solution based on fuzzy decision

With the Pareto-optimal set of nondominated solutions, the proposed approach presents the power system decision maker with the best compromise solution. A fuzzy-based mechanism extracts the best compromise solution over the trade-off curve and helps the decision maker efficiently adjust generation levels [23]. The decision maker's judgement is imprecise, so each objective function of the i^{th} solution is a membership function μ_i defined as flow:

$$\mu_i = \begin{cases} 1 & F_i \le \min(F_i) \\ \frac{\max(F_i) - F_i}{\max(F_i) - \min(F_i)} & \min(F_i) \leqslant F_i \le \max(F_i), \\ 0 & F_i \ge \max(F_i) \end{cases} \qquad (24)$$

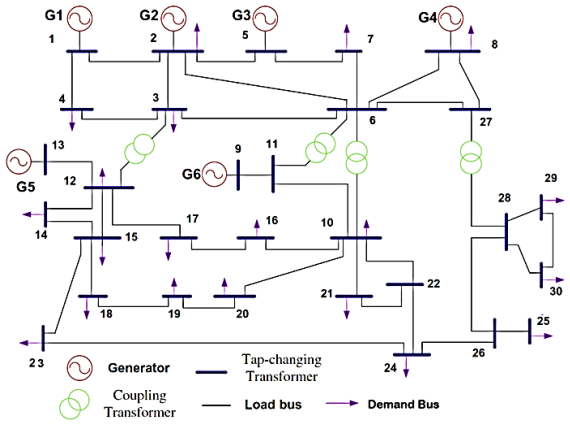
where min (Fi) and max (Fi) are lower and upper bounds of ith objective function. The higher the values of the membership function are, the greater the solution satisfaction is. For each nondominated solution, the normalized membership function μ^k is calculated as:

$$\mu^k = \frac{\sum_{i=1}^{N_{obj}} \mu_i^k}{\sum_{k=1}^{M} \sum_{i=1}^{N_{obj}} \mu_i^k}, \qquad (25)$$

where M is the number of nondominated solutions, and N_{obj} is the number of objects. The best compromise solution is the one having the maximum of μ^k.

5.2 Simulation Results: The IEEE 30-bus system in Fig. 9 is used to test NS-MOABC, MOPSO, MOABC, and NSGA-II to verify the proposed approach. The IEEE 30-bus test case represents a Midwestern American Electric Power System segment in December 1961. IEEE stands for Institute of Electrical and Electronics Engineers. Six generators, 41 transmission lines, and 4 transformers with off-nominal tap ratio in lines 6–9, 6–10, 4–12, and 27–28 comprise the IEEE 30-bus system. System data are in [55]. The active power generation limits are in Table 9. Generator buses and load buses have limits of 0.95–1.1 and 0.9–1.05 p.u. The transformer tap step size is 0.01 p.u., and the lower and upper limits are 0.9 and 1.05 p.u.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024



Generator	G_1	G_2	G_3	G_4	G_5	G_6
Pc_{max} (MW)	150	150	150	150	150	150
Pc_{min} (MW)	5	5	5	5	5	5
Cost coefficients						
a	10	10	20	10	20	10
b	200	150	180	100	180	150
c	100	120	40	60	40	100
Emission coefficients						
α	4.091	2.543	4.258	5.326	4.258	6.131
β	-5.554	-6.047	-5.094	-3.550	-5.094	-5.555
γ	6.490	5.638	4.586	3.380	4.586	5.151
ξ	2.0e-4	5.0e-4	1.0e-6	2.0e-3	1.0e-6	1.0e-5
λ	2.857	3.333	8.000	2.000	8.000	6.667

Table 7: Characteristics of the generation units

Figure 9: Network structure of IEEE 30-bus system and it's parameters

5.2.1: Case I: two-objective OPF optimization

As a multi-objective optimisation problem, the OPF model optimises each two objective functions simultaneously. Figs. 10–12 show the Pareto fronts obtained by the NS-MOABC, MOPSO, MOABC, and NSGA-II algorithms for cost–emission, loss–cost, and loss–emission pairs. Tables 10–15 show the two-dimensional Pareto front's Pareto-optimal and best compromise solutions for each objective. All results on three two-objective OPF problems show that the proposed NS-MOABC algorithm can obtain well distributed Pareto-optimal fronts. The proposed approach achieves the trade-off between competing objectives by emphasising non-dominated solutions and a well-distributed set of solutions.

MOOPF	NS-MOABC		MOABC		NSGA-II		MOPSO	
	Best f1	*Best f2*	*Best f1*	*Best f2*	*Best f1*	*Best f2*	*Best f1*	*Best f2*
P_{G1}	12.34	41.25	10.34	6.71	36.17	41.08	26.06	55.72
P_{G2}	30.18	45.38	30.14	41.46	55.31	48.60	33.32	46.15
P_{G3}	55.42	55.22	57.45	61.66	49.53	51.19	61.58	55.92
P_{G4}	105.21	41.37	105.49	100.81	46.07	43.09	101.88	40.22
P_{G5}	45.27	52.82	45.36	41.51	56.81	51.95	45.17	55.89
P_{G6}	37.85	51.59	37.35	33.95	44.11	51.77	37.84	49.87
f_1 Fuel cost ($/h)	606.66	645.69	606.67	608.80	643.74	645.93	607.08	645.88
f_2 Emission (ton/h)	0.2240	0.1912	0.2261	0.2241	0.1965	0.1953	0.2249	0.1952

Table 8: The best solutions for cost and emission (f_1–f_2) from the Pareto front of MOOPF

MOOPF	NS-MOABC	MOABC	NSGA-II	MOPSO
P_{G1}	24.4937	25.1875	36.1782	24.2855
P_{G2}	39.1558	39.4798	55.3158	37.6143
P_{G3}	56.9355	38.5551	49.5340	55.3510
P_{G4}	71.1990	74.1921	46.0786	72.0443
P_{G5}	50.5460	48.0866	56.8116	51.5658
P_{G6}	44.6195	43.0481	44.1181	44.5503
f_1 Fuel cost ($/h)	**616.8721**	615.5057	643.7436	616.6645
f_2 Emission (ton/h)	**0.2033**	0.2035	0.1965	0.2024

Table 9: The best compromise solutions for cost and emission (f_1–f_2) of MOOPF

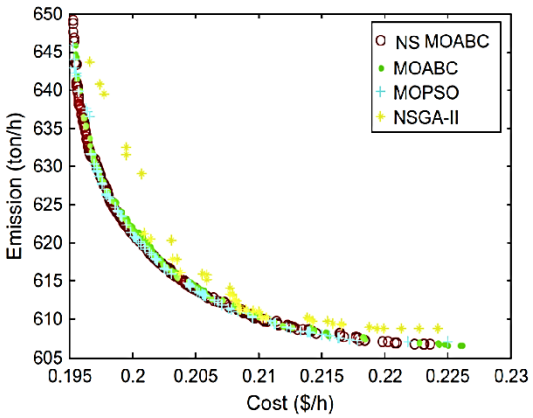


Figure 10: Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II for cost and emission (f_1–f_2).

Table 8 shows that NS-MOABC converges best for fuel cost and emission objective functions. Fig. 10 shows fuel cost and emission objective functions' Pareto-optimal front. Fig. 11 shows that NS-MOABC finds a well-distributed and diverse solution set for this problem, while the other three algorithms only find sparse distributions. The best compromise fuel cost and emission objective functions using different algorithms are shown in Table 9.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

MOOPF	NS-MOABC		MOABC		NSGA-II		MOPSO	
	Best f_1	*Best f_3*	*Best f_1*	*Best f_3*	*Best f_1*	*Best f_3*	*Best f_1*	*Best f_3*
P_{G1}	15.02	2.27	9.77	18.36	3.91	2.19	67.12	8.88
P_{G2}	29.32	26.33	33.91	20.81	10.63	23.71	38.31	9.20
P_{G3}	56.27	106.50	60.90	40.30	110.74	73.77	68.44	108.80
P_{G4}	112.58	56.87	95.39	119.33	67.45	98.03	101.65	70.32
P_{G5}	30.22	5.08	45.29	40.49	10.96	5.00	37.02	72.36
P_{G6}	42.28	88.71	40.88	46.70	81.33	82.29	28.28	33.63
f_1 (Fuel cost $/h)	608.53	664.76	607.4267	611.67	656.80	637.30	609.33	660.21
f_3 (Loss MW)	2.2923	1.6089	2.7710	2.6197	1.6407	1.6186	2.6137	2.2421

Table 10: The best solutions for cost and loss (f_1–f_3) from the Pareto front of MOOPF.

MOOPF	NS-MOABC	MOABC	NSGA-II	MOPSO
P_{G1}	4.4166	8.5482	3.1622	11.5906
P_{G2}	24.5609	28.8145	12.4072	20.6855
P_{G3}	80.6696	75.3171	87.9835	83.0087
P_{G4}	106.0433	105.2485	92.0489	96.5241
P_{G5}	7.7227	17.4874	27.8416	11.8301
P_{G6}	61.6627	49.8543	61.8298	71.89055
f_1 Fuel cost ($/h)	**1.6754**	1.8702	1.873572	1.62575
f_3 Loss (MW)	**622.5278**	613.3222	623.2921	630.6447

Table 11: The best compromise solutions for cost and loss (f_1–f_3) of MOOPF.

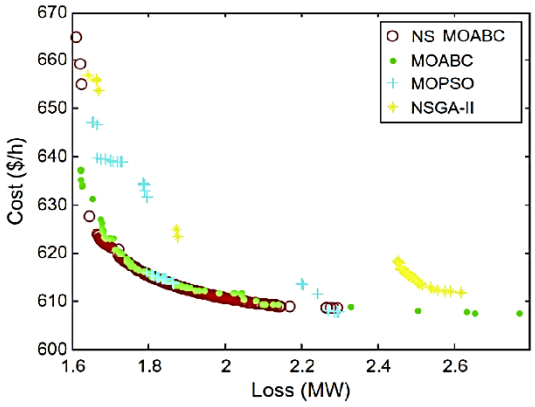


Figure 12: Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II for cost and loss (f_1–f_3).

The best compromise for NS-MOABC is 616.8721 $/h and 0.2033 ton/h. If the best compromise solution's fuel cost and emission values are close to Table 10, then the statement is true in all Pareto dimensions. The algorithms rank similarly for cost–loss and emission–loss objective functions as for fuel cost–emission objective functions.

MOOPF	NS-MOABC		MOABC		NSGA-II		MOPSO	
	Best f_2	*Best f_3*	*Best f_2*	*Best f_3*	*Best f_2*	*Best f_3*	*Best f_2*	*Best f_3*
P_{G1}	45.36	9.11	43.96	33.86	11.95	63.28	32.92	11.48
P_{G2}	44.83	5.00	41.84	59.34	6.50	144.46	44.71	76.36
P_{G3}	56.45	90.97	57.31	50.04	69.67	108.61	62.17	23.07
P_{G4}	40.30	71.16	40.65	34.62	95.30	61.97	42.82	63.32
P_{G5}	46.60	5.00	42.59	63.52	6.41	5.17	50.07	82.53
P_{G6}	53.86	103.75	34.62	47.13	95.19	34.62	52.75	58.56
f_2 Emission (ton/h)	0.1956	0.2592	0.2182	0.1974	0.2523	0.2732	0.1959	0.2823
f_3 Loss (MW)	4.0313	1.6026	4.6123	5.1511	1.6475	2.1242	3.8294	1.7634

Table 12: The best solutions for emission and loss (f_2–f_3) from the Pareto front of MOOPF

MOOPF	NS-MOABC	MOABC	NSGA-II	MOPSO
P_{G1}	20.8113	69.3327	30.4071	22.4024
P_{G2}	33.6987	22.0615	32.0927	25.0679
P_{G3}	77.5603	31.1704	76.1658	102.4471
P_{G4}	65.1139	66.8127	66.2653	63.9250
P_{G5}	17.4145	77.6187	33.3860	6.4725
P_{G6}	70.7318	18.4182	47.7291	86.2670
f_2 Emission (ton/h)	**2.0198**	2.0144	2.6463	1.5965
f_3 Loss (MW)	**0.2238**	0.2134	0.2051	0.2445

Table 13: The best compromise solutions for emission and loss (f_2–f_3) of MOOPF

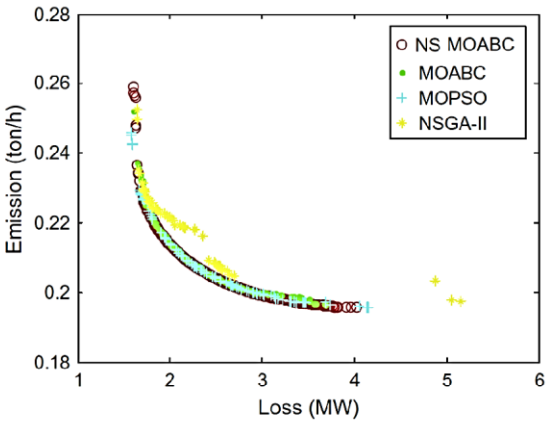


Figure 13: Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II for emission and loss (f_2–f_3).

5.2.2: Case II: three-objective OPF optimization: The proposed algorithm optimises three competing objectives simultaneously Cost, emission, and loss cannot be improved without compromising the other two optimised objectives. Choose the best implementation compromise from the Pareto-optimal solutions. Tables 14 show that NS-MOABC outperforms all other algorithms in best and compromise fuel cost, emission, and loss solutions. Fig. 14 shows that the NS-MOABC finds a well-distributed and diverse solution set for this three-objective problem. The other three algorithms cannot archive the three-objective OPF's true Pareto front. The results again show that the proposed method provides a well-distributed Pareto-optimal front for three-objective OPF optimisation. The results show that the NS-MOABC algorithm solves the real-world multi-objective OPF problem with multiple Pareto-optimal solutions in one run.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 11, No.3, November : 2024

MOOPF	NS-MOABC	MOABC	NSGA-II	MOPSO
P_{G1}	18.7228	18.4350	36.1782	21.2740
P_{G2}	32.9912	28.1939	55.3158	14.7574
P_{G3}	67.4162	73.5849	49.5340	89.6146
P_{G4}	83.1738	86.8403	46.0786	87.0153
P_{G5}	31.9749	26.1992	56.8116	7.1517
P_{G6}	51.4947	52.2758	44.1181	65.3095
f_1 Fuel cost ($/h)	**613.1135**	613.9435	623.1181	631.3939
f_2 Emission (ton/h)	**0.2118**	0.2169	0.2022	0.2333
f_3 Loss (MW)	**2.3739**	2.1293	3.2935	1.7228

Table 14: The BCS for cost, emission and loss (f_1–f_2–f_3) of MOOPF

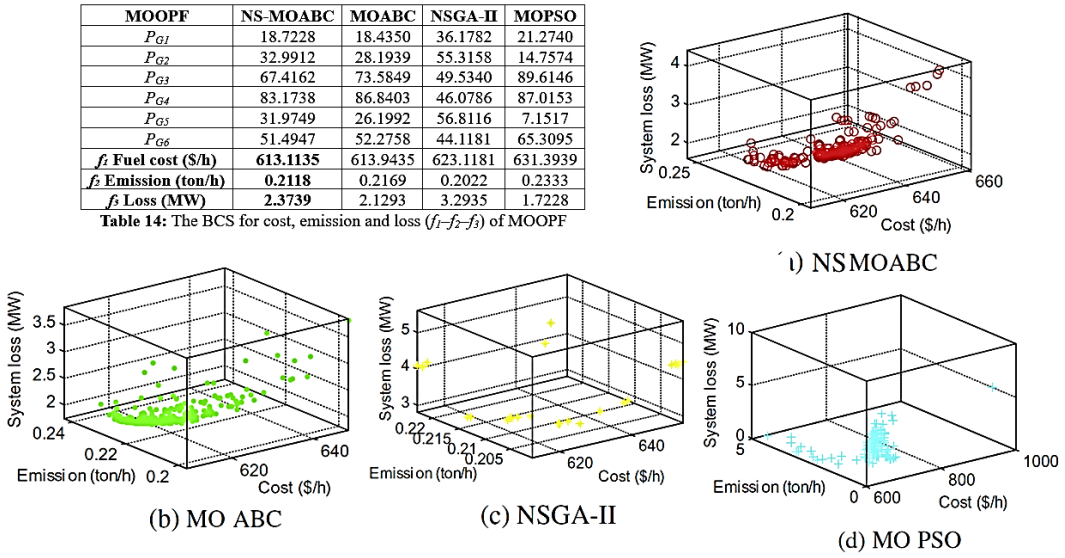


Figure 14**:** Pareto fronts obtained by NS-MOABC, MOPSO, MOABC, and NSGA-II for fuel cost, emission and loss (f_1–f_2–f_3).

6 Statistical test: To test the efficacy and robustness of the proposed NS-MOABC on the real-world OPF problem, the analysis of variance (ANOVA) test was used to determine the statistical significance of each of the four algorithms. Visual statistics are analysed using box plots. Box plots are great for visualising distribution details. The middle half of the distribution's scores are in the box, which runs from the 25th percentile to the 75th percentile. The median line spans the box. Thus, one-fourth of the distribution is between this line and the box's top and bottom.

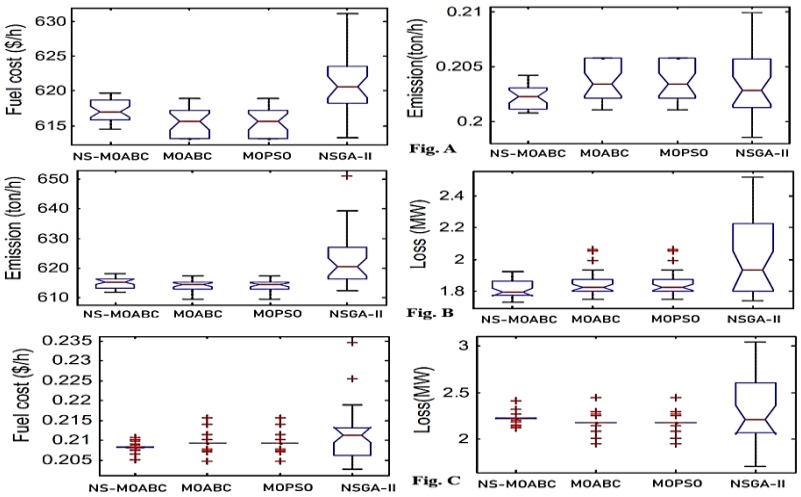


Figure 15: The box plots of best compromise solutions obtained by NS-MOABC, MOABC, MOPSO and NSGA-II for 30 runs (A) For fuel cost and emission (f_1–f_2) pairs (B) for fuel cost and loss (f_1–f_3) pairs (C) for emission and loss (f_2–f_3) pairs

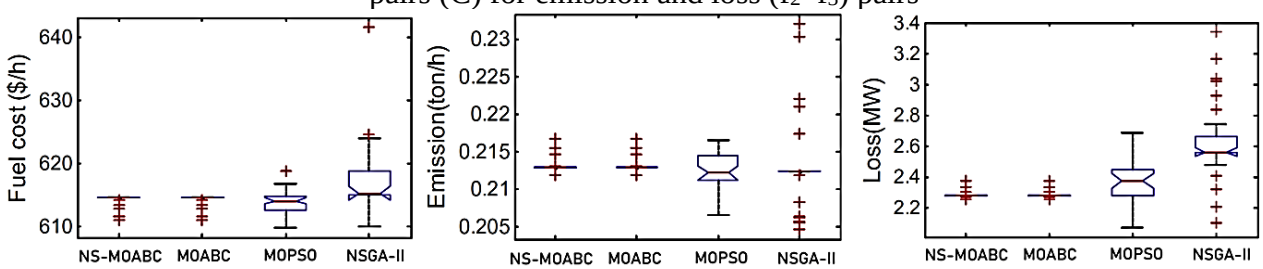


Figure 16: The box plots of best compromise solutions obtained by NS-MOABC, MOABC, MOPSO and NSGA-II for fuel cost, emission and loss (f_1–f_2–f_3) pairs

Figs.15–16 show the ANOVA test results for all algorithms on two and three objectives in 30 runs. The distribution's general characteristics can be seen in these box plots. Fig. 23 shows the box plots for Table 11's best fuel cost and emission compromise solutions (f_1–f_2). NS-MOABC has the best variance of compromise solutions with fuel costs between 615 and 620, as shown in Fig. 24.

Computation time analysis: Real-world optimisation problems require high optimisation accuracy, computation robustness, and solution speed. Thus, this experiment included computation time analysis to verify the NS-MOABC's efficacy. The time spent on each generation varies by algorithm, so iterations do not represent time spent. In Fig. 17, we showed the computing times (average in 30 runs) of all algorithms on each multi-objective OPF case to easily evaluate algorithmic time response. Fig. 17 shows that NS-MOABC takes the longest to compute. The other algorithms took less computing time to optimise, but NS-MOABC took fewer iterations to achieve more robust and precise OPF solutions. Thus, reducing maximum iterations saves NS-MOABC run time.

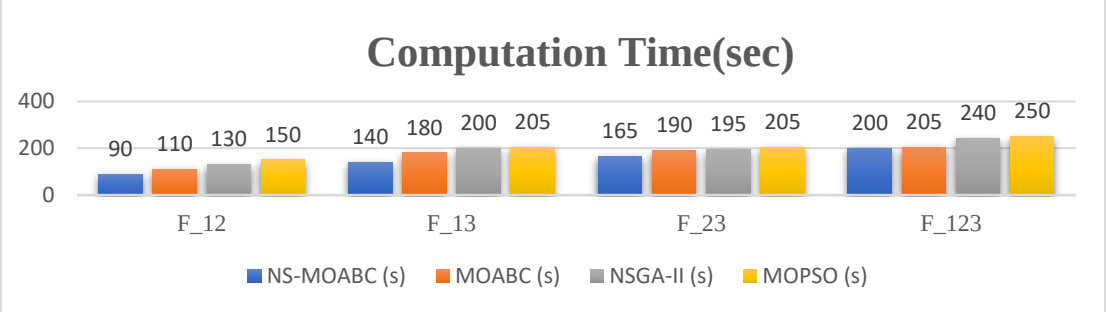


Figure 17: Computation Time of MOOPF State-of-the-art of Algorithms.

III. Conclusion

This paper introduces NS-MOABC, a multi-objective optimisation algorithm inspired by bees' intelligent foraging. This method uses the Pareto concept, external archive, elite selection, crowding distance strategies, and adaptive learning to converge to the true Pareto optimal front while maintaining population diversity through non-dominated sorting. A new adaptive search procedure in the bee phase guides solutions to the optimal region while preserving diversity, refined by a fitness strategy based on Pareto rank and crowding distance. NS-MOABC performed well on numerical benchmark problems, so the authors applied it to multi-criteria power flow optimisation, considering cost, loss, and emission. The simulation results show that NS-MOABC can solve OPF problems better than MOPSO, MOABC, and NSGA-II. The algorithm's simplicity and diversity preservation make it useful for complex multi-objective optimisation problems. NS-MOABC outperforms MOPSO, MOABC, and NSGA-II in six mathematical benchmark functions for two and three objectives. The proposed multi-criteria power flow optimisation model's best compromise solution is found using fuzzy membership. Compared to MOPSO, MOABC, and NSGA-II, NS-MOABC had better optimisation accuracy and convergence robustness in 30-bus IEEE test system simulations.

References

[1] Avadh Kishor, Pramod Kumar Singh, and Jay Prakash, "NSABC: Non-dominated sorting based multi-objective artificial bee colony algorithm and its application in data clustering," in Proceedings of Neurocomputing, Elsevier, 2016, pp. 514–533.

[2] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization," in Proceedings of the First IEEE Conference on Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence, IEEE, 1994, pp. 82–87.

[3] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," Evolutionary Computation, vol. 2, no. 3, pp. 221–248, 1994.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Transactions on Evol. Computation, vol. 6, no. 2, pp. 182–197, 2002.

[5] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," IEEE Transactions on Evol. Com., vol. 3, no. 4, pp. 257–271, 1999.
[6] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," IEEE Transactions on Evol. Computation, vol. 8, no. 3, pp. 256–279, 2004.
[7] G. Xu, Y.-Q. Yang, B.-B. Liu, Y.-H. Xu, and A.-J. Wu, "An efficient hybrid multi-objective particle swarm optimization with a multi-objective dichotomy line search," Journal of Computational and Applied Mathematics, vol. 280, pp. 310–326, 2015.
[8] S. Omkar, D. Mudigere, G. N. Naik, and S. Gopalakrishnan, "Vector evaluated particle swarm optimization (VEPSO) for multi-objective design optimization of composite structures," Computers & Structures, vol. 86, no. 1, pp. 1–14, 2008.
[9] C. Horoba, and F. Neumann, "Approximating Pareto-optimal sets using diversity strategies in evolutionary multi-objective optimization," in Advances in Multi-Objective Nature Inspired Computing, Springer, 2010, pp. 23–44.
[10] X. Zou, M. Liu, L. Kang, and J. He, "A high performance multi-objective evolutionary algorithm based on the principles of thermodynamics," in Parallel Problem Solving from Nature - PPSN VIII, Springer, 2004, pp. 922–931.
[11] X. Zou, Y. Chen, M. Liu, and L. Kang, "A new evolutionary algorithm for solving many-objective optimization problems," IEEE Transactions on Cybernetics: Part B, vol. 38, pp. 1402–1412, 2008.
[12] H. Sato, H. E. Aguirre, and K. Tanaka, "Controlling dominance area of solutions and its impact on the performance of MOEAs," in Evolutionary Optimization, Springer, 2007, pp. 5–20.
[13] Yen GG, Lu H. Dynamic multiobjective evolutionary algorithm: adaptive cellbased rank and density estimation. IEEE Trans Evol Comput 2003;7:253–74.
[14] Surender Reddy S, Bijwe PR, Abhyankar AR. Faster evolutionary algorithm based optimal power flow using incremental variables. Int J Electr Power Energy Syst 2014; 54:198–210.
[15] Rezaei Adaryani M, Karami A. Artificial bee colony algorithm for solving multiobjective optimal power flow problem. Int J Electr Power Energy Syst 2013; 53:219–30.
[16] Lai LL, Ma JT, Yokoyama R, Zhao M. Improved genetic algorithm for optimal power flow under both normal and contingent operation states. Electr Power Energy Syst 1997;19:287–92.
[17] Gao W, Liu S. Improved artificial bee colony algorithm for global optimization. 2011;871–82.
[18] Karaboga D, Akay B. A comparative study of ABC. Appl Math Comput 2009;214:108–32.
[19] Omkar SN, Senthilnath J, Khandelwal Rahul, Narayana Naik G, Gopalakrishnan S. Artificial bee colony for multi-objective optimization of composite structures. Appl Soft Comput 2011:489–99.
[20] Hedayatzadeh R, Hasanizadeh B, Akbari R, Ziarati K. A multi-objective artificial bee colony for optimizing multi-objective problems. In: 3rd international conference on advanced computer theory and engineering (ICACTE), Chengdu, China; 2010. p. 277–81.
[21] Qu BY, Suganthan PN. Multi-objective evolutionary algorithms based on the summation of normalized objectives and diversified selection. Inf Sci 2010;180(17):3170–81.
[22] Jensen MT. Reducing the run-time complexity of multiobjective EAs: the NSGA-II and other algorithms. IEEE Trans Evol Comput 2003;7:503–15.