Enhancing Moss Growth Optimization with Genetic Algorithm for Improved Performance

Kaushal Thakre; Purva Landge; Sandhya Dahake
Department of Master in Computer Application
G H Raisoni College of Engineering & Management Nagpur,
Maharashtra, India

Abstract: This study proposes MGO-GA, a hybrid optimization approach that combines Genetic Algorithm (GA) with Moss Growth Optimization (MGO). In complex search environments, MGO often suffers from premature convergence and stagnation, despite its great exploration and exploitation capabilities. The suggested approach improves population variety and speeds up convergence by introducing genetic operators crossover and mutation into the MGO process to get around these problems. By using GA's global search ability, the hybrid MGO-GA enhances the results generated by MGO and reduces the possibility of being trapped in local optima.

Keywords: Hybridization, Benchmark functions, Convergence, Exploration and Exploitation.

1. Introduction

Moss Growth Optimization (MGO) is a nature-based metaheuristic algorithm which known for its strong exploration and exploitation abilities [1]. However, like many other optimization techniques, MGO suffered from premature convergence and stagnation, restricting its effectiveness in complex search spaces [2]. These challenges raised due to a lack of variations in candidate solutions, reducing the MGO's ability to escape local optima [4]. To overcome these limitations, a hybrid MGO-GA (Moss Growth Optimization with Genetic Algorithm) approach was proposed, integrating genetic operators such as crossover and mutation to enhance diversity and improve convergence speed [3]. The solutions generated by MGO are refined with GA's global search capability, reducing the risk of getting trapped in local optima [6]. Experimental results on benchmark functions showed that MGO hybridized with GA achieved better accuracy, stability and convergence rate as compared to standard MGO algorithm, making it more reliable and efficient optimization approach.

2. Literature Review

2.1 Selection of Moss Growth Optimization (MGO):

Moss growth optimization was a newly developed nature inspired meta heuristic algorithm that mimics the adaptive growth waiver of moss it effectively balances the exploration (searching new areas in the solution space) and exploitation (refining the best solutions found so far) [8]. MGO was chosen for this study due to its strong optimization capabilities and efficient exploration-exploitation balance. Its promising results in various benchmark functions further justified its selection [6].

2.2 Justification for Hybridization with Genetic Algorithm (GA):

GA was selected for hybridization with MGO due to its strong global search capability and genetic operators, which enhance solution diversity and prevent premature convergence [9]. By incorporating crossover and mutation, GA improves exploration, helping MGO escape local optima and achieve better optimization performance [10].

DOI: https://doi.org/10.5281/zenodo.15571838

2.3 Classification of Optimization Algorithms:

Algorithms are generally classified into natural, genetic, human-based, and revolutionary. Natural algorithms are inspired by biological processes [1]. Genetic algorithms are based on the principle of natural selection [3]. Human-based algorithms leverage human intelligence [4], and Revolutionary algorithms introduce innovative computational methods [5].

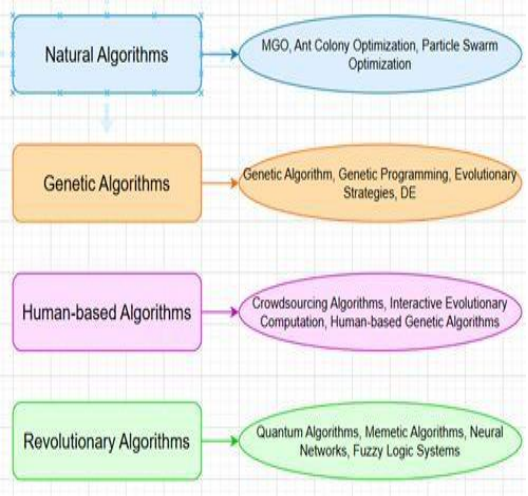


Fig 1. Nature-inspired algorithm classification

2.4 Algorithms and Authors:

The Table1 represents various previously developed nature-inspired algorithms. These algorithms are used in solving complex optimization problems by mimicking natural phenomena, biological behaviors, or physical principles.

Table 1: Algorithm, Authors & Year of publishing

Sr. No.	Algorithm Name	Author Name	Year
1.	Ant Colony Optimization (ACO)	Marco Dorigo	1992
2.	Grey Wolf Optimizer (GWO)	Seyedali Mirjalili et al	2014
3.	Moss Growth Optimization (MGO)	Ali Asghar Heidari et al	2024
4.	Genetic Algorithm (GA)	Jhom Holland	1975
5.	Differential Evolution	Rainer Storn et al	1997
6	Particle Swarm Optimization	James Kennedy et al	1995
7.	Sine Cosine Algorithm	Seyedali Mirjalili	2016

2.5 Pseudo Code:

The Hybrid MGO-GA algorithm combines Moss Growth Optimization (MGO) with a Genetic Algorithm (GA) to enhance solution accuracy and convergence speed. MGO explores the search space, and GA refines the best solution using selection, crossover, and mutation. The convergence curves of both algorithms are merged for performance analysis. This hybrid approach improves optimization efficiency by leveraging MGO's exploration and GA's genetic diversity.

2.5.1 Code:

Function Hybrid_MGO_GA(N, MaxFEs, lb, ub, dim, fobj):
Initialize random seed for reproducibility
// Step 1: Run MGO Algorithm
Print "Running MGO..."
Call MGO(N, MaxFEs, lb, ub, dim, fobj)
Store results in Best_pos_MGO, Best_score_MGO, Convergence_curve_MGO
Print Best_score_MGO
// Step 2: Run GA for Fine-Tuning MGO's Best Solution
Print "Refining with GA..."
Set GA_Iterations to 100
Call GA(Best_pos_MGO, lb, ub, dim, fobj, GA_Iterations)
Store results in Best_pos, Best_score, Convergence_curve_GA
// Step 3: Merge Convergence Data
Find minimum length of Convergence_curve_MGO and Convergence_curve_GA
Resize both convergence curves to this minimum length
Convert both to column vectors
Concatenate the two curves side-by-side into Convergence_curve
Print Best_score
Function GA(initial_pos, lb, ub, dim, fobj, maxGen):
Set population size to 20
Set mutation rate to 0.1
Set crossover rate to 0.8
Initialize population around initial_pos
Apply boundary constraints to population

DOI: https://doi.org/10.5281/zenodo.15571838

Evaluate fitness of population
Store best score and best position
Initialize Convergence_curve
For each generation in maxGen:
Perform tournament selection to choose parents
Perform crossover to generate offspring
Perform mutation on offspring
Evaluate fitness of offspring
Replace worst individuals in population with better offspring
Update best score and best position if a better solution is found
Store best score in Convergence_curve
Function tournamentSelection(population, fitness, popSize):
Initialize parents array
For each individual in population:
Randomly select two individuals
Choose the one with better fitness as a parent
Return parents array
Function crossover(parents, crossoverRate, lb, ub):
Initialize offspring array as parents
For each pair of parents:
If random number is less than crossoverRate:
Perform simulated binary crossover (SBX) to generate offspring
Apply boundary constraints to offspring
Return offspring
Function mutation(offspring, mutationRate, lb, ub):
Initialize mutated array as offspring
For each individual in offspring:
If random number is less than mutationRate:
Randomly select a dimension
Perform random mutation within bounds
Return mutated array
Function replaceWorst(population, fitness, offspring, offspring_fitness):
Combine population and offspring
Combine fitness and offspring fitness
Sort combined fitness
Select the top individuals to form new population
Return new population and new fitness

2.6 Functions and Equations

Benchmark functions are standardized mathematical models that are used to evaluate optimization algorithms. Various search space complexities, including unimodal, multimodal, and non-convex landscapes, are represented to assess an algorithm's efficiency in exploration and convergence. The ability of an algorithm to escape local optima and find the global optimum in diverse optimization scenarios is tested using these functions. Smooth, rugged, and deceptive landscapes are included to ensure a comprehensive performance evaluation.

Table 2: Standard UM Benchmark Functions

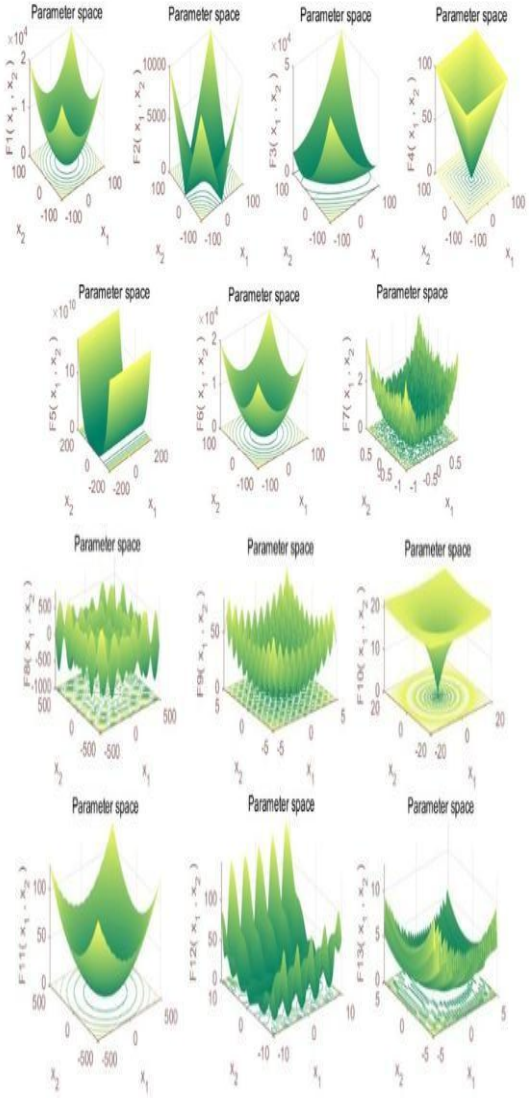
Functions	Dimensions	Range	f_{min}
$F_1(S) = \sum_{m=1}^{z} S_m^2$	(10,30,50,100)	[-100 , 100]	0
$F_2(S) = \sum_{m=1}^{z} \|S_m\| + \prod_{m=1}^{z} \|S_m\|$	(10,30,50,100)	[-10 ,10]	0
$F_3(S) = \sum_{m=1}^{z} (\sum_{n=1}^{m} S_n)^2$	(10,30,50,100)	[-100 , 100]	0
$F_4(S) = max_m\{\|S_m\|, 1 \le m \le z\}$	(10,30,50,100)	[-100 , 100]	0
$F_5(S) = \sum_{m=1}^{z-1} [100(S_{m+1} - S_m^2)^2 + (S_m - 1)^2]$	(10,30,50,100)	[-38 , 38]	0
$F_6(S) = \sum_{m=1}^{z} ([S_m + 0.5])^2$	(10,30,50,100)	[-100 , 100]	0
$F_7(S) = \sum_{m=1}^{z} mS_m^4 + random[0,1]$	(10,30,50,100)	[-1.28, 1.28]	0
$F_8(S) = \sum_{m=1}^{z} -S_m sin(\sqrt{\|S_m\|})$	(10,30,50,100)	[-500,500]	-418.98295
$F_9(S) = \sum_{m=1}^{z} [S_m^2 - 10cos(2\pi S_m) + 10]$	(10,30,50,100)	[-5.12,5.12]	0
$F_{10}(S) = -20exp\left(-0.2\sqrt{\left(\frac{1}{z}\sum_{m=1}^{z} S_m^2\right)}\right) - exp\left(\frac{1}{z}\sum_{m=1}^{z} cos(2\pi S_m)\right) + 20 + d$	(10,30,50,100)	[-32,32]	0
$F_{11}(S) = 1 + \sum_{m=1}^{z} \frac{S_m^2}{4000} - \Pi_{m=1}^{z} cos\frac{S_m}{\sqrt{m}}$	(10,30,50,100)	[-600, 600]	0
$F_{12}(S) = \frac{\pi}{z}\{10sin(\pi\tau_1) + \sum_{m=1}^{z-1}(\tau_m - 1)^2[1 + 10sin^2(\pi\tau_{m+1})] + (\tau_z - 1)^2\} + \sum_{m=1}^{z} u(S_m, 10,100,4)$ $\tau_m = 1 + \frac{S_m + 1}{4}$ $u(S_m, b, x, i) = \begin{cases} x(S_m - b)^i & S_m > b \\ 0 & -b < S_m < b \\ x(-S_m - b)^i & S_m < -b \end{cases}$	(10,30,50,100)	[-50,50]	0
$F_{13}(S) = 0.1\{sin^2(3\pi S_m) + \sum_{m=1}^{z}(S_m - 1)^2[1 + sin^2(3\pi S_m + 1)] + (x_z - 1)^2[1 + sin^2 2\pi S_z)]$	(10,30,50,100)	[-50,50]	0
$F_{14}(S) = \left[\frac{1}{500} + \sum_{n=1}^{2} 5\frac{1}{n + \sum_{m=1}^{2}(S_m - b_{mn})^6}\right]^{-1}$	2	[-65.536, 65.536]	1
$F_{15}(S) = \sum_{m=1}^{11} \left[b_m - \frac{S_1(a_m^2 + a_m S_2)}{a_m^2 + a_m S_3 + S_4}\right]^2$	4	[-5, 5]	0.00030
$F_{16}(S) = 4S_1^2 - 2.1S_1^4 + \frac{1}{3}S_1^6 + S_1S_2 - 4S_2^2 + 4S_2^4$	2	[-5, 5]	-1.0316
$F_{17}(S) = (S_2 - \frac{5.1}{4\pi^2}S_1^2 + \frac{5}{\pi}S_1 - 6)^2 + 10(1 - \frac{1}{8\pi})cosS_1 + 10$	2	[-5, 5]	0.398
$F_{18}(S) = \left[1 + (S_1 + S_2 + 1)^2(19 - 14S_1 + 3S_1^2 - 14S_2 + 6S_1S_2 + 3S_2^2)\right] \times \left[30 + (2S_1 - 3S_2)^2(18 - 32S_1 + 12S_1^2 + 48S_2 - 36S_1S_2 + 27S_2^2)\right]$	2	[-2,2]	3
$F_{19}(S) = -\sum_{m=1}^{4} d_m exp\left(-\sum_{n=1}^{3} S_{mn}(S_m - q_{mn})^2\right)$	3	[1, 3]	-3.32
$F_{20}(S) = -\sum_{m=1}^{4} d_m exp\left(-\sum_{n=1}^{6} S_{mn}(S_m - q_{mn})^2\right)$	6	[0, 1]	-3.32
$F_{21}(S) = -\sum_{m=1}^{5} [(S - b_m)(S - b_m)^T + d_m]^{-1}$	4	[0,10]	-10.1532

DOI: https://doi.org/10.5281/zenodo.15571838

$F_{22}(S) = -\sum_{m=1}^{7}[(S-b_m)(S-b_m)^T+d_m]^{-1}$	4	[0, 10]	-10.4028
$F_{23}(S) = -\sum_{m=1}^{7}[(S-b_m)(S-b_m)^T+d_m]^{-1}$	4	[0, 10]	-10.5363

2.6 Search Space

The search space illustrated in the benchmark functions represents diverse optimization landscapes, including unimodal, multimodal, convex, and non-convex surfaces. These functions define the complexity of the optimization problem, influencing an algorithm's ability to explore the space, escape local optima, and converge toward the global optimum.



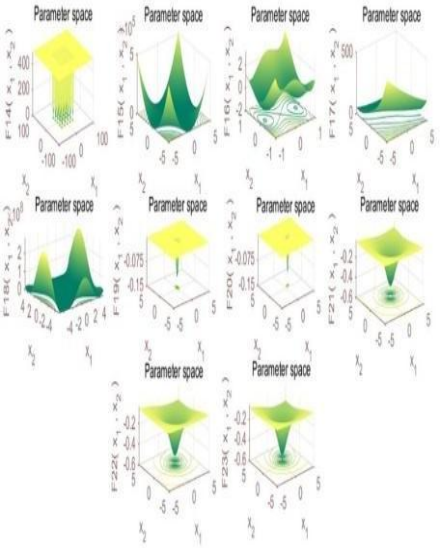


Fig 2. 23 Benchmark Functions

3. Result & Discussion

The performance of the proposed MGO-GA hybrid approach was evaluated on 23 benchmark functions and compared against the standard Moss Growth Optimization (MGO) algorithm. The results indicate that MGO was outperformed by MGO-GA in 14 out of 23 functions (~61% improvement), demonstrating that solutions were effectively refined and optimization performance was enhanced.

Table 3: Result for 23 Benchmark Function

Function Name	Best Score of MGO	Best Score of MGO-GA
F1	8.2824	6.9672
F2	0.40466	0.70836
F3	8946.4473	8281.6172
F4	27.3275	25.9857
F5	2757.8191	1793.5856
F6	5.7435	4.7054
F7	0.049169	0.029917
F8	-11132.832	-11793.2468
F9	36.9431	25.2683
F10	1.2067	25.2683
F11	1.0763	1.0718
F12	1.5441	0.61047

DOI: https://doi.org/10.5281/zenodo.15571838

F13	**5.8758**	**4.182**
F14	**0.998**	**0.998**
F15	**0.00079911**	**0.00077964**
F16	**-1.0316**	**-1.0316**
F17	**0.39789**	**0.39955**
F18	**3**	**3.0162**
F19	**-3.8628**	**-3.8626**
F20	**-3.3195**	**-3.309**
F21	**-10.1406**	**-10.1455**
F22	**-10.3868**	**-10.4029**
F23	**-10.5314**	**-10.5355**

4. Conclusion

The performance of the proposed MGO-GA hybrid approach was evaluated across 23 benchmark functions, and comparative analysis revealed improvements in 14 of them, specifically in functions F1, F3, F4, F5, F6, F7, F8, F9, F11, F12, F13, F15, F21, and F22. This conclusion was drawn based on metrics such as solution accuracy, convergence speed, and consistency across multiple runs. However, certain functions namely F2, F10, and F19 exhibited minimal improvement, with F10 showing a noticeable performance drop. These results indicate that integrating GA operators into MGO enhances the algorithm's effectiveness in many scenarios.

5. Refrences

[1] B. Zheng, Y. Chen, C. Wang, A. A. Heidari, L. Liu, and H. Chen, "Moss Growth Optimization: Concepts and Performance," *Journal of Computational Design and Engineering*, 2024, doi: 10.1093/jcde/qwae080.

[2] A. A. Heidari, R. Ali Abbaspour, and A. Rezaee Jordehi, "An efficient chaotic water cycle algorithm for optimization tasks," *Neural Comput. Appl.*, vol. 28, no. 1, pp. 57–85, 2017, doi: 10.1007/s00521-015-2037-2.

[3] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley, 1989.

[5] M. Mafarja et al., "Evolutionary Population Dynamics and Grasshopper Optimization for Feature Selection," *Knowledge-Based Syst.*, vol. 145, pp. 25–45, 2018, doi: 10.1016/j.knosys.2017.12.037.

[6] G. Venter and J. Sobieszczanski-Sobieski, "A Genetic Algorithm for Constrained Optimization Problems," *AIAA J.*, vol. 40, no. 7, pp. 1379-1385, 2002.

[7] Y. Liang, A. Törn, and M. Viitanen, "Benchmark Functions for Global Optimization," *Springer Optim. Appl.*, vol. 30, pp. 123-150, 2009

[8] A. A. Heidari, R. Ali Abbaspour, and A. Rezaee Jordehi, "An efficient chaotic water cycle algorithm for optimization tasks," Neural Comput. Appl., vol. 28, no. 1, pp. 57–85, 2017, doi: 10.1007/s00521-015-2037-2.

[9] R. V. Rao and G. G. Waghmare, "A new optimization algorithm for solving complex constrained design optimization problems," vol. 0273, no. April, 2016, doi: 10.1080/0305215X.2016.1164855.

[10] R. Al-Hajj and A. Assi, "Estimating solar irradiance using genetic programming technique and meteorological records," AIMS Energy, 2017, doi: 10.3934/energy.2017.5.798.

[11] W. Y. Lin, "A novel 3D fruit fly optimization algorithm and its applications in economics," Neural Comput. Appl., 2016, doi: 10.1007/s00521-015-1942-8.

[12] Y. Cheng, S. Zhao, B. Cheng, S. Hou, Y. Shi, and J. Chen, "Modeling and optimization for collaborative business process towards IoT applications," Mob. Inf. Syst., 2018, doi: 10.1155/2018/9174568.

[13] X. Wang, T. M. Choi, H. Liu, and X. Yue, "A novel hybrid ant colony optimization algorithm for emergency transportation problems during post-disaster scenarios," IEEE Trans. Syst. Man, Cybern. Syst., 2018, doi: 10.1109/TSMC.2016.2606440.

[14]I. E. Grossmann, Global Optimization in Engineering Design (Nonconvex Optimization and Its Applications), vol. 9. 1996.

DOI: https://doi.org/10.5281/zenodo.15571838

[15] E.-S. M. El-Kenawy, M. M. Eid, M. Saber, and A. Ibrahim, "MbGWO-SFS: Modified Binary Grey Wolf Optimizer Based on Stochastic Fractal Search for Feature Selection," IEEE Access, 2020, doi: 10.1109/access.2020.3001151.

DOI: https://doi.org/10.5281/zenodo.15571838