SLUJST
Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

A Study of Hybridized Smell Agent Symbiotic Organism Search in Congress on Evolutionary Computation Functions

Salisu Mohammed[1], Yusuf A. Sha'aban[2], Ime J. Umoh[3], Ahmed T. Salawudeen[4], Ibrahim O. Muritala[5]
[1]Department of Maintenance Engineering, Nigerian National Petroleum Company, KRPC Ltd, Kaduna, Nigeria.
[2]Department of Electrical Engineering, Faculty of Engineering, University of Hafr Al Batin, KSA.
[3,5]Department of Computer Engineering, Faculty of Engineering, Ahmadu Bello University, Zaria, Nigeria.
[4]Department of Electrical and Electronics Engineering, Faculty of Engineering, University of Jos, Nigeria.
salisu.mohammed3@nnpcgroup.com[1], shaaban@uhb.edu.sa[2], ijumoh@abu.edu.ng[3], atsalawudeen@unijos.edu.ng[4], drolawalemi@gmail.com[5]

Abstract

This paper presents a study of the Smell Agent Symbiotic Organism Search (SASOS) hybrid algorithm. SASOS is developed from bioinspired Smell Agent-Based Optimization (SAO) and Symbiosis Organism Search (SOS) algorithms. Bioinspired algorithms often lack a balance between speed and accuracy to achieve optimal performance efficiency and a global search for the best solution. To address these challenges, the algorithm reduces the imbalance between diversification and intensification in bioinspired algorithms to improve the search for global optima. SASOS performance was evaluated in sixteen selected Congress on Evolutionary Computation (CEC) functions using Aggregative Best Counts (ABC) compared to the regular SAO and SOS algorithms. For an advanced performance comparison, the convergence study was carried out on each CEC function to assess the fitness of the algorithms based on the Desirable Convergence Goal (DCG). Evaluation results using 50 iterations have shown that SASOS performed better with ABC of 56.25% than the SAO and SOS algorithms with ABC of 28.12% and 15.63%, respectively, in standard benchmark functions. Furthermore, in the convergence study, 1000 iterations were superimposed for each algorithm on the CEC functions. The convergence results showed that SASOS obtained the best DCG of 58.83% compared to SOS and SAO with DCG of 25.00% and 16.67%, respectively. These results made the performance of the hybrid SASOS uniquely different from other similar approaches. This is because the hybrid SASOS satisfactorily balanced the diversification and intensification phases in the bioinspired SAO and SOS algorithms. The eligible characteristics of the hybrid SASOS with respect to ABC and DCG showed its compatibility and significance for various engineering optimization applications.

Keywords: *Bioinspired, Diversification, Hybrid Algorithm, Intensification, Optimization*

1. Introduction

Hybrid algorithms are essential for advancing optimization concepts (Zellagui *et al.*, 2021). Optimization entails the application of computed algorithms to solve a problem with a specific objective function and bounded constraints (Mohammed *et al.*, 2022). For more than a few decades, optimization algorithms have been vital in addressing various challenges affecting humanity, technology, and the economic environment (Radosavljević *et al.*, 2015). However, many algorithms are designed based on specific problems, which limits their optimal performance in different applications (Yu *et al.*, 2019). Modern optimization tools include evolutionary and bioinspired algorithms (Lang & Jia, 2019). The latter reflects an advanced optimization level due to their natural inspiration, easy computation, and implementation strategies (Kesavan *et al.*, 2019). Most of the time, evolutionary algorithms seek the best solution, but with insufficient computational time. As such, the algorithms do not guarantee an optimal solution in the required time (Mokarram *et al.*, 2019). Other disadvantages of evolutionary algorithms include early convergence and trapping in local minima (Goldanloo & Gharehchopogh, 2022). In the contrast, many bioinspired algorithms apply a specific strategy to avoid local optima. As a result, these algorithms can solve complex optimization problems without trapping in local optima (Zhang *et al.*, 2022). Approaches to bioinspired algorithms include mimics

Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

of physical creatures and swarm intelligence (Pani, 2021). These approaches successfully solve constrained multi-objective optimization problems (Abdullahi *et al.*, 2020). However, according to the Pareto principle of optimality, a single algorithm cannot optimally solve all multi-objective constrained optimization problems simultaneously, but rather a set of compromises in performance (Abdullahi *et al.*, 2016). Although single algorithms can find optimal solutions, their execution time increases exponentially with increasing dimensionality of the problem (Kang *et al.*, 2023). For example, some algorithms have greater diversification (exploration), whereas others have greater intensification (exploitation) in the search for global optima (Salawudeen *et al.*, 2021). Intensification is the convergence ability in bioinspired algorithms to locate optimal or suboptimal solutions within the search space as quickly as possible. Diversification is the ability of bioinspired algorithms to navigate the whole search space to obtain the best of the optimal solution (Abdullahi *et al.*, 2020). Consequently, greater diversification often causes sluggish performance, while more significant intensification leads to a suboptimal solution (Goldanloo & Gharehchopogh, 2022). The goal of any algorithm to obtain the best optimal solution requires a balance between the diversification and intensification phases in the search environment (Vinaykumar *et al.*, 2023). The search for an optimal solution is carried out continuously in a number of iterations until the most suitable solution relevant to the problem is obtained in a reasonable time frame (Sayed *et al.*, 2023). As a result, two or more algorithms are hybridized to enhance the performance weakness of one another (Kang *et al.*, 2023).

The remainder of this paper is organized as follows. Related works in the literature with a theoretical review describing the fundamental functions of SOS, SAO, and CEC are detailed in Section 2. The development and implementation sequence of hybrid SASOS is presented in Section 3. The results and discussion with convergence analysis of the algorithms compared in the CEC functions are presented in Section 4. The conclusions and recommendations are presented in Section 5.

2. Related Works

This section reviews some related work in the literature. Several strategies in hybrid optimization algorithms have been developed and implemented. Some examples of such hybridized algorithms include Ant Colony Optimizer and Genetic Algorithm (ACO-GA) (Zukhri & Paputungan, 2013); Artificial Bee Colony and Ant Colony Optimizer (ABC-ACO) (Sen & Mathur, 2016); Cuckoo Search and Particle Swarm Optimizer (CS-PSO) (Guo *et al.*, 2016); Grey Wolf and Whale Optimizer (GW-WO) (Singh & Hachimi, 2018); Thermal Exchange and Seagull Optimizer (TE-SO) algorithm (Jia et al., 2019); Salpa-Swarm Optimizer and Backpropagation Learning (SSO-BL) algorithm (Pani, 2021); Aquila Optimizer and Arithmetic Optimizer (AO-AO) algorithm (Zhang *et al.*, 2022); Symbiosis Organism Search and Improved Firefly (SOS-IF) algorithm (Goldanloo & Gharehchopogh, 2022); Neighborhood Variable Search and Improved Differential Evolutionary (NVS-IDE) algorithm (Kang *et al.*, 2023); Equilibrium Optimizer and Teaching-Learning Based Optimizer (EO-TLBO) algorithm (Sayed *et al.*, 2023).

In general, various hybrid algorithms developed address optimization strategies, which are rather tricky to handle by a single algorithm (Al-Tameemi *et al.*, 2022; Tao *et al.*, 2022). Among the hybridized algorithms, ACO-GA updates and preserves the fitness functions in each optimization cycle, simplifying computational complexity to find the best optimal solution. This applied hybrid algorithm obtained the shortest route to the travelling salesman problem compared to applied single algorithms (Zukhri & Paputungan, 2013). The ABC-ACO hybrid finds initial solutions to improve probable solutions and discards inferior solutions using the replacement method. This improves the chances of a globally optimal solution (Sen & Mathur, 2016). The developed CS-PSO hybrid solves a cost function using an optimized finite-time search, improving the convergence and searchability of an optimal solution. The applied algorithm obtained an optimal solution to the periodic maintenance scheduling problem (Guo *et al.*, 2016). The GW-WO hybrid rapidly resolves the imbalance between the exploration and exploitation phases. This algorithm avoids trapping in local optima by quickly updating the constrained functions when tested in pressure vessels and welded beam designs (Singh & Hachimi, 2018). The TE-SO hybrid algorithm balances intensification and diversification that improves high accuracy in task classification, which improves the performance speed in processor units (Jia et al., 2019). The SSO-BL hybrid algorithm enhances the accuracy of data in steganographic techniques, which improves big data security performance in the domain of Internet of Things (IoT). A highly secure IoT protocol was transferred to the destination using secure steganography (Pani, 2021). The AO-AO hybrid algorithm was developed to enhance the equilibrium between exploitation and exploration by minimizing the randomness of the energy parameters in each

Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

algorithm (Zhang et al., 2022). The SOS-IF hybrid algorithm was developed to accelerate convergence towards optimal solution to the continuous optimization problem. The firefly optimizer was improved by an oppositional-based learning technique, which hybridizes SOS to improve the efficiency of exploitation and exploration performance. Validation of the algorithm on benchmarks revealed that it has an efficient solution compared to other algorithms (Goldanloo & Gharehchopogh, 2022). The NVS-IDE hybrid algorithm was developed as a two-stage optimizer (TSO) to provide an optimal solution to scheduling problems. The algorithm minimizes the maximum completion time of the scheduled tasks by solving different batches of the quantity allocation problem. The TSO based NVS-IDE showed better performance effectiveness than the single-stage optimizer algorithm (Kang et al., 2023). The EO-TLBO hybrid algorithm negotiates high-exploitation, high-exploration at the equilibrium point in global search space. The algorithm successfully tuned the gains of the PI controller, improving the performance of the PV system under dynamic irradiance. Performance comparison showed the efficacy of EO-TLBO over conventional standalone algorithms (Sayed et al., 2023). Other hybrid algorithms have been reported in the literature, such as in Mafarja & Mirjalili (2017), Jia *et al.* (2019), Jafari *et al.,* (2020), and Vinaykumar *et al.,* (2023). This paper hybridizes Smell Agent-Based Optimization (SAO) and Symbiotic Organism Search (SOS) due to their specific bioinspiration and natural similarities

2.1 Theoretical Review

This section presents the theoretical review, material and methods for developing a hybrid algorithm. Materials used include a computer system and associated installed software programs, including MATLAB/Simulink R2020b. These materials were used in the program design to develop the SASOS algorithm and implement it in selected CEC functions. A fundamental review of the SOS, SAO and CEC functions used in the design application is carried out. The sequence chart, algorithm steps for developing hybrid SASOS, and implementation of the algorithms are presented in detail. The simulation results and discussions including convergence analysis and performance comparison of the algorithms on CEC functions are elucidated. Finally, the conclusion is presented with recommendations.

2.1.1 Symbiotic organism search

Symbiotic Organism Search (SOS) applies symbiosis characteristics between living organisms interacting in the ecosystem (Abdullahi *et al.,* 2016). The organisms interact in facultative or obliged states. The facultative state implies that the organisms mutually cohabitate to benefit in a non-essential relationship (Shankar & Mukherjee, 2016). The oblige condition means that the survival of both organisms strictly depends on each other (Abdullahi *et al.,* 2020). In SOS, the symbiosis relationship has three distinctive phases: mutuality, commensality, and parasitism. These phases are described as follows (Cheng & Prayogo, 2014; Abdullahi *et al.,* 2016; Hasanien & El-Fergany, 2017; Saha & Mukherjee, 2018):

1. Mutuality: In this phase, the organisms interchange the symbiotic advantages for survival in the ecosystem. The variable vectors, x_i and x_j are randomly deployed to represent the mutual organism in the search space to increase the chances of survival. The survival chance for each organism is considered its benefit factor (BF) (Cheng & Prayogo, 2014). In the mutuality process, the fitness functions of x_i and x_j are upgraded by the best degree of survival chances known as x_{best}. The process regenerates new organisms as expressed in (1) and (2):

$$x_{inew}^{m}=x_i+\text{rand}\,(\cdot)*\left(x_{best}-\left(\left(x_i+x_j\right)/2\right)*BF_1\right) \quad (1)$$

$$x_{jnew}^{m}=x_j+\text{rand}\,(\cdot)*\left(x_{best}-\left(\left(x_i+x_j\right)/2\right)*BF_2\right) \quad (2)$$

where, x_{inew}^{m} and x_{jnew}^{m} are the new organisms regenerated, x_{best} is a vector representing organism with the best survival chance, BF_1 and BF_2 are the benefit factors with values determined randomly between -1 and 1. The term $\left(x_i+x_j\right)/2$ is a mutuality vector of the organisms.

2. Commensality: In this phase, only one of the organisms benefits from the symbiotic interaction without affecting the other (Abdullahi *et al.,* 2016). A vector x_i is selected to randomly interrelates with a vector x_j so that only x_i benefits. The vector x_j neither gains nor loose in the interaction (Cheng & Prayogo, 2014). The fitness function of

Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

x_i is upgraded by the x_{best} organism using a set of randomly generated numbers to produce a new commensality vector in (3) (Abdullahi *et al.*, 2020):

$$x_{inew}^{c}=x_{inew}^{m}+\text{rand}\,(\cdot)*\left(x_{best}-x_{jnew}^{m}\right) \quad (3)$$

3. Parasitism: In this phase, the organisms involved interrelate so that one benefits and the other remains harmed. In the relationship between x_i and x_j, a parasitic vector x_i^p is generated to mutualize with x_i. If the fitness function in vector, x_i^p is better, x_j will die and its portion will be occupied by x_i. If the immunity in fitness function of x_j is higher, x_i^p will die in the ecosystem (Hasanien & El-Fergany, 2017).

2.1.2 Smell agent-based optimization

Smell Agent-based Optimization (SAO) applies principles of combinatory optimization using a sense of smell by an agent. The projected molecules of smell are evaluated by an agent, which tracks and obtains the best solution to optimization problem. For the best tracking and detection of these molecules, SAO applies three strategic modes described as follows (Salawudeen *et al.*, 2019):

i. Sniff mode: The extent to which an agent perceives the smelling molecules projected from the source.
ii. Trail mode: Trackability of an agent to detect the smelling projected molecules.
iii. Random mode: This irregular search strategy allows the agent to optimize the shortest path to detect the smelling molecules projected from the source.

The mathematical modelling of SAO is based on the listed modes. The sniffing expression is given in (4) as follows (Tijani *et al.*, 2018):

$$X_{i\_s}^{(t+1)}=X_i^{(t)}+V_i^{(t+1)} \quad (4)$$

where $X_i^{(t+1)}$ and $X_i^{(t)}$ represent the currently and previous positions of the molecules, respectively. $V_i^{(t)}$ denote the velocity of the molecule. The upgraded sniffing position is expressed in (5).

$$X_{i\_s}^{(t+1)}=X_i^{(t)}+\left(V_i^{(t)}+r_0\sqrt{\frac{3KT}{m}}\right) \quad (5)$$

where r_0 defines the random number, m represents mass of the molecule, K denotes the Boltzmann constant, and T defines the temperature level of the molecule (Salawudeen *et al.*, 2019). The modelling of trailing mode is expressed in (6):

$$X_{i\_t}^{(t+1)}=X_i^{(t)}+r_1(\cdot)*f_{olc}*\left(X_{agent}^{(t)}-X_i^{(t)}\right)-r_2(.)*f_{olc}*\left(X_{worst}^{(t)}-X_j^{(t)}\right) \quad (6)$$

where, $X_{agent}^{(t)}$ and $X_{worst}^{(t)}$ represent vectors of the trailing agent and worst organism, respectively. The $r_1(.)$ and $r_2(.)$ are random numbers generated at each time interval, f_{olc} is the olfaction capability (Salawudeen *et al.*, 2019). The random mode is given in (7) as follows (Salawudeen *et al.*, 2021):

$$X_i^{(t+1)}=X_i^{(t)}+r_3(\cdot)*SM \quad (7)$$

where, SM is the step movement, and $r_3(.)$ is the random number penalty (Salawudeen *et al.*, 2019). The combinatory action of the SAO modes gives an optimal global solution to the optimization problem (Salawudeen *et al.*, 2021). The SAO and SOS algorithms are selected to develop the SASOS algorithm. The algorithm developed applies the strategic smell sensibility of an agent in the search for a symbiotically beneficial relationship in the ecosystem. The trailing

Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

mode of SAO introduces a control parameter, while the SOS phases determine the best mutual interaction for an optimal solution in the hybrid SASOS algorithm.

2.1.3 The CEC functions

The CEC functions evaluate the performance of any hybridized, modified, or newly developed algorithm (Shankar & Mukherjee, 2016). Functions are structured using one or more contours in the subsets of mathematical representations. The goal is to validate the algorithm's integrity under test by minimizing or maximizing the contours. This is achieved using various equality or inequality bounds corresponding to the algorithm's characteristics, dimension, and range. In this paper, 16 strategically selected multidimensional CEC functions in Table 1. are used as benchmarks.

Table 1: Multidimensional CEC functions (Shankar & Mukherjee, 2016)

F/No.	F/Name	D	Formula	Range	G. min.
Fn1	Beale	2	$f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$	$[-4.5,4.5]$	0
Fn2	Easom	2	$f(x_1, x_2) = -\cos(x_1)\cos(x_2)\exp(-(x_1 - \pi)^2 - ((x_1 - \pi)^2))$	$[-100,100]$	-1
Fn3	Matyas	2	$f(x) = 0.26(x_1^2 + x_1^2) - 0.48x_1x_2$	$[-10,10]$	0
Fn4	Bohanchevsky1	2	$f(x) = \sum_{i=1}^{n-1}[x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i) - 0.4\cos(4\pi x_{i+1}) + 0.7]$	$[-100,100]$	0
Fn5	Booth	2	$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	$[-10,10]$	0
Fn6	Michalewicz2	2	$f(x) = -\sum_{i=1}^{2}\sin(x_i)(\sin(ix_i^2/\pi))^{20}$	$[0,\pi]$	-1.8013
Fn7	Schaffer	2	$f(x)\sum_{i=1}^{30}((x_i^2 + x_{i+1}^2)^{0.25})\{[\sin 50(x_i^2 + x_{i+1}^2)^{0.1}]^2 + 1\}$	$[-100,100]$	0
Fn8	Six Hump Camelback	2	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$[-5,5]$	-1.03163
Fn9	Bohachevsky 2	2	$f(x) = \sum_{i=1}^{n-1}[x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i)(4\pi x_{i+1}) + 0.3]$	$[-100,100]$	0
Fn10	Shubert	2	$f(x) = \left(\sum_{i=1}^{n} i\cos(i+1)x_i + i\right)\left(\sum_{i=1}^{n} i\cos(i+1)x_{i+1} + i\right)$	$[-10,10]$	-186.73
Fn11	Colville	4	$f(x) = 100(x_1 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2) + 19.8(x_2 - 1)(x_4 - 1)$	$[-10,10]$	0
Fn12	Michalewicz5	5	$f(x) = -\sum_{i=1}^{2}\sin(x_i)(\sin(ix_i^2/\pi))^{20}$	$[0,\pi]$	-4.6877
Fn13	Zakharov	10	$f(x) = \sum_{i=1}^{D}x_i^2 + \left(\sum_{i=1}^{D}0.5ix_i\right)^2 + \left(\sum_{i=1}^{D}0.5ix_i\right)^4$	$[-5,10]$	0
Fn14	Michalewicz10	10	$f(x) = -\sum_{i=1}^{2}\sin(x_i)(\sin(ix_i^2/\pi))^{20}$	$[0,\pi]$	-9.6602
Fn15	Step	30	$f(x) = \sum_{i=1}^{n}(x_i + 0.5)^2$	$[-5.12,5.12]$	0

Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

Fn16	Sphere	30	$f(x) = \sum_{i=1}^{D} x_i^2$	$[-100,100]$	0

The multimodal and unimodal functions with different characteristics, ranges, and dimensions are shown in Table 1. The functions represent various degrees of the optimization spectrum. The SASOS algorithm is evaluated using these functions and compared with the SAO and SOS algorithms in Section 3.

3. Development of Smell Agent Symbiosis Organism Search Algorithm

This section discusses the development of the hybrid SASOS algorithm. SASOS is developed on codification of the trailing mode of SAO with phases of SOS. The algorithm performs the optimization procedure using program loops consisting of mutuality, commensality, and the introduced trailing agent. The optimization sequence in the SASOS algorithm is detailed in the flow chart shown in Figure 1

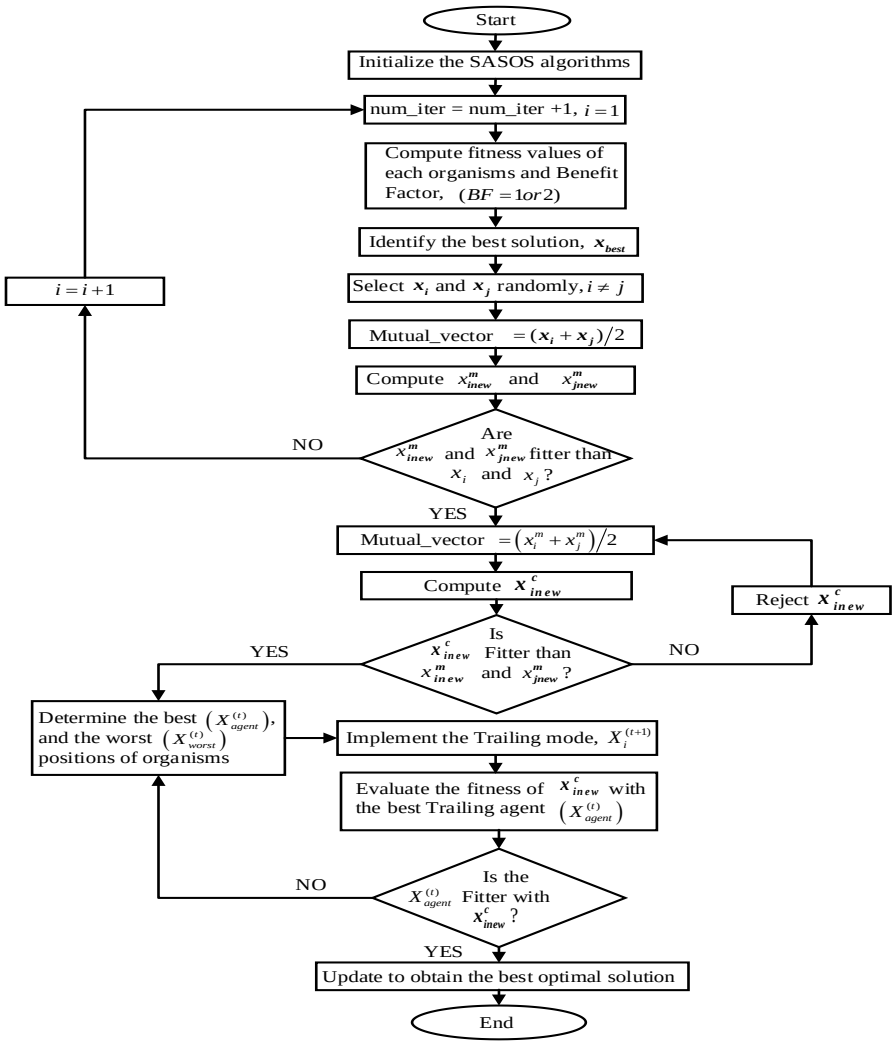


Figure 1. Flow chart of smell agent symbiotic organism search

The flow chart of the SASOS algorithm is shown in Figure 1. The overall program is initialized as the SASOS algorithm. The first iteration began by computing fitness values and benefit factors for the ecosystem's organism. Next, the mutuality vectors are computed to generate two new organism vectors, which produce a strong commensality

Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

vector. This vector is evaluated to provide a better value in determining the best and worst positions. If the commensality vector is not fitter than the two new mutuality vectors, it is rejected, and the program returns to recompute the mutuality vectors. The trailing mode is activated based on the input provided by the best and worst positions of the organisms. The fitness of the best trailing agent is then evaluated. If the best agent vector is fitter with the new commensality vector, the trailing agent is upgraded to obtain the best optimal solution. Otherwise, the program goes back to the beginning of trailing mode. The SASOS optimization are detailed in Algorithm 1.

Algorithm 1: Smell Agent Symbiotic Organism Search Algorithm

1. Initialize population of organisms, x_i and x_j, set eco-size and stopping criteria.
2. Determine the maximum iteration number.
3. Obtain the best organism, x_{best}, and the best trailing agent, $X_{agent}^{(t)}$.
4. Compute the mutuality vector of organisms, x_i and x_j.
5. Compute the commensality vector, x_i^c.
6. Determine the best $X_{agent}^{(t)}$, and the worst $X_{worst}^{(t)}$ organisms' positions.
7. Implement the trailing mode, $X_{i\_t}^{(t+1)}$.
8. Evaluate the fitness of x_i^c with the trailing agent, $X_{agent}^{(t)}$.
9. Apply the fittest $X_{agent}^{(t)}$ with x_i^c to obtain the optimal desired solution.
10. Repeat 2 to 9 until the stopping criteria are reached.

3.1 Implementation of Optimization Algorithms

The developed SASOS, standard SAO and SOS algorithms are implemented on the CEC functions. The functions are programmed to prompt the selection of the assigned serial number, n for each function. For smooth running of the selected algorithm, the assigned function is evaluated to provide an optimal solution within the search space. The flow chart of the implementation process is shown in Figure 2.

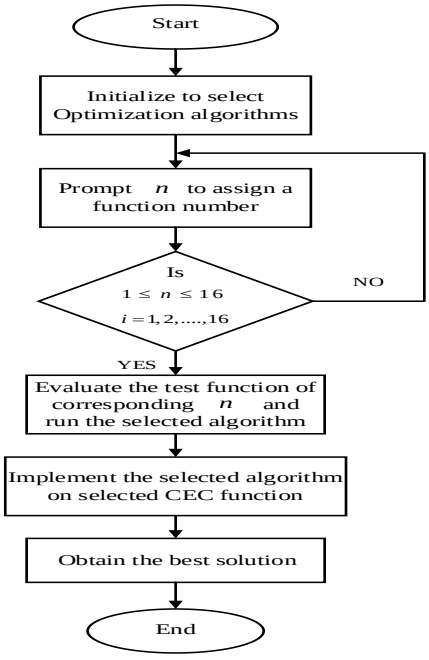


Figure 2. Flow chart for implementation of optimization algorithms

A flow chart for implementing the optimization algorithms is shown in Figure 2. The process starts by initiating a selected optimization algorithm, and then a function number n is prompted to input the corresponding function

SLUJST
Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

number. A verification decision is made such that if the assigned value of n_i is within an acceptable range, the corresponding function is evaluated to run with the selected algorithm. Otherwise, the program goes back to prompt n until the correct function is assigned. The best solution obtained is shown with the corresponding time of convergence.

4. Results and Discussion

In this section, the simulation results are presented for performance evaluation and comparison. To determine the optimization integrity of the SASOS algorithm in 16 CEC functions, fifty independent iterations were conducted for each algorithm on different functions. Multidimensional subsets of the functions were used to verify the performance of the SASOS. The results are compared with regular SOS and SAO algorithms. The best solutions, convergences, averages, and standard deviations obtained are presented and compared in Table 3. The table elucidates the results obtained in the 2D, 4D, 5D, 10D, and 30D selected functions. Values in bold indicate the best values obtained by the corresponding algorithm. Italic values indicate a situation where two or more algorithms obtained similar best values. The italic bold values obtained in the shortest time are considered the best compared to similar non-bold italic values. The ranking of each algorithm is decided from the statistical counts of the best values in bold. The Aggregative Best Counts (ABC) in Table 3 summarize the performance comparison of the algorithms presented in Table 2.

Table 2. Comparison of algorithms on multidimensional CEC functions

F/No	F/Name	Characteristics	Dimensions	Performance	G. Min.	SAO	SOS	SASOS
F_{n1}	Beale	Multimodal, non-separable	2D	Conv Time (s)	0.00	8.6645	6.6496	**4.1514**
				Best Solution		*0.00*	*0.00*	***0.00***
				Average		2.8734*e*-5	*0.00*	***0.00***
				Std Dev		1.4219*e*-4	*0.00*	***0.00***
F_{n2}	Easom	Multimodal, non-separable	2D	Conv Time (s)	-1.00	8.1186	5.5268	**2.0487**
				Best Solution		-0.9999	*-1.00*	***-1.00***
				Average		-0.0400	*-1.00*	***-1.00***
				Std Dev		0.1979	*0.00*	***0.00***
F_{n3}	Matyas	Unimodal, non-separable	2D	Conv Time (s)	0.00	7.2655	2.7820	**2.6909**
				Best Solution		**0.00**	2.6176*e*-36	5.3838*e*-35
				Average		7.0452*e*-9	6.1080*e*-18	**1.1448*e*-18**
				Std Dev		3.4865*e*-8	1.8329*e*-17	**4.2996*e*-18**
F_{n4}	Bohachevsky1	Multimodal, non-separable	2D	Conv Time (s)	0.00	8.1536	**5.5929**	5.7295
				Best Solution		7.8863*e*-4	***0.00***	*0.00*
				Average		5.5603*e*-6	***0.00***	*0.00*
				Std Dev		2.7516*e*-5	***0.00***	*0.00*
F_{n5}	Booth	Unimodal, separable	2D	Conv Time (s)	0.00	7.4940	**1.9055**	3.1199
				Best Solution		2.8116*e*-5	5.1052*e*-20	**1.6237*e*-23**
				Average		1.6937*e*-4	1.2011*e*-17	**1.1274*e*-17**

SLUJST
Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

						8.3817*e*-5	**7.1907*e*-18**	1.7406*e*-17
F_{n6}	Michalewicz2	Multimodal, non-separable	2D	Conv Time (s)		**6.2613**	6.9287	6.4347
				Best Solution	-1.8013	***-1.8013***	*-1.8013*	*-1.8013*
				Average		-0.0720	*-1.8013*	***-1.8013***
				Std Dev		0.3563	**5.4277*e*-8**	7.3031*e*-7
F_{n7}	Schaffer	Multimodal, non-convex	2D	Conv Time (s)		7.8380	2.5932	**2.0206**
				Best Solution	0.00	*0.00*	*0.00*	***0.00***
				Average		0.0120	2.2204*e*-18	**0.00**
				Std Dev		0.0597	1.0988*e*-17	**0.00**
F_{n8}	Six Hump Camel Back	Multimodal, non-separable	2D	Conv Time (s)		**5.3026**	6.8689	5.4952
				Best Solution	-1.0316	***-1.0316***	*-1.0316*	*-1.0316*
				Average		***-1.0316***	*-1.0316*	*-1.0316*
				Std Dev		**4.3026*e*-15**	7.9295*e*-15	4.3026*e*-15
F_{n9}	Bohachevsky2	Multimodal, non-separable	2D	Conv Time (s)		7.6979	3.4790	**2.1145**
				Best Solution	0.00	*0.00*	*0.00*	***0.00***
				Average		2.8968*e*-6	**0.00**	4.4408*e*-18
				Std Dev		1.4336*e*-5	**0.00**	1.5212*e*-17
F_{n10}	Shubert	Multimodal, separable	2D	Conv Time (s)		10.5151	**3.1287**	3.8639
				Best Solution	-186.73	-186.7108	**-186.7302**	-186.7009
				Average		-7.4683	-	-
				Std Dev		6.9584*e*5	**186.7306**	186.7009
							3.3760*e*-4	**2.8710*e*-14**
F_{n11}	Coville	Multimodal, continuous	4D	Conv Time (s)		8.9279	4.6818	**4.5185**
				Best Solution	0.00	1.2778*e*-17	3.4975*e*-17	**0.00**
				Average		0.3636	**1.0752*e*-12**	3.4824*e*-12
				Std Dev		1.7995	**7.6022*e*-12**	2.4624*e*-11
F_{n12}	Michalewicz5	Multimodal, separable	5D	Conv Time (s)		23.4407	**2.4260**	9.0004
				Best Solution	-4.6877	-4.5325	***-4.6877***	*-4.6877*
				Average		**-4.6877**	-4.6862	-4.6843
				Std Dev		**-0.0114**	2.5354	0.6993
F_{n13}	Zakharov	Unimodal, non-separable	10D	Conv Time (s)		12.7956	2.8119	**2.2147**
				Best Solution	0.00	5.5353*e*-26	8.9476*e*-27	**0.00**
				Average		2.3349*e*-5	7.2220*e*-18	**6.1362*e*-18**
				Std Dev		1.1555*e*-4	2.1261*e*-17	**1.6973*e*-17**
F_{n14}	Michalewicz10	Multimodal, non-separable	10D	Conv Time (s)		**28.2766**	77.0582	70.5448
				Best Solution	-9.6602	-5.4417	*-9.6602*	***-9.6602***
				Average		-0.2177	*-9.6602*	***-9.6602***
				Std Dev		1.0772	1.1328*e*6	***0.2281***

Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

F/No	F/Name	Characteristics	Dimensions	Performance	G. Min.	SAO	SOS	SASOS
F_{n15}	Step	Multimodal, non-separable	30D	Conv Time (s) Best Solution Average Std Dev	0.00	15.7264 1.4684*e*-20 8.5868*e*-5 4.2494*e*-4	**2.2812** 1.3838*e*-20 1.1169*e*-17 **2.1249*e*-17**	2.0738 **0.00** **1.0538*e*-17** 2.1790*e*-17
F_{n16}	Sphere	Unimodal, separable	30D	Conv Time (s) Best Solution Average Std Dev	0.00	16.2483 5.2448*e*-19 4.7057*e*-6 2.3287*e*-5	2.6999 7.1593*e*-30 4.5166*e*-18 1.5997*e*-17	**2.5689** **8.6155*e*-31** **2.4476*e*-18** **9.4460*e*-18**
Counts of Best Convergence.						**3**	**5**	**8**
Counts of Best Solution.						**3**	**3**	**10**
Counts of Best Average.						**2**	**4**	**10**
Counts of Best Std. Dev.						**2**	**6**	**8**
Performance Ranks.						**3**	**2**	**1**

Table 3. Aggregative best count performance comparison

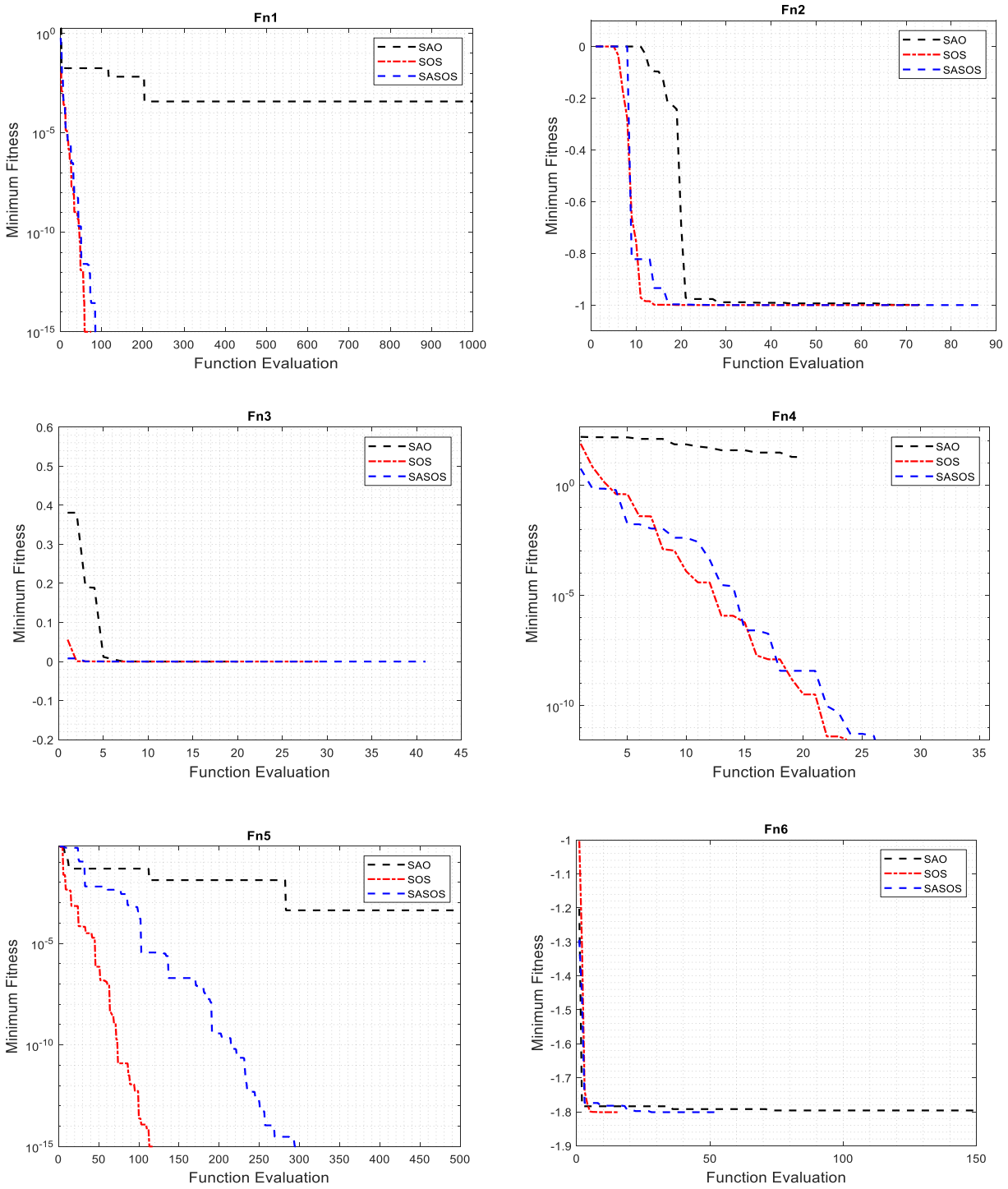
Algorithms.	Performance	2D	4D	5D	10D	30D	Aggregate.	% Aggregate.	Rank.
SASOS	Conv Time (s).	5	1	-	1	1	8	50.00%	
	Best Solution.	5	1	-	2	2	10	62.50%	
	Average.	6	-	-	2	2	10	62.50%	
	Std. Dev.	5	-	-	2	1	8	50.00%	
	Aggregate	21	2	-	7	6	**36**	**56.25%**	**1**
SOS	Conv Time (s).	3	-	1	-	1	5	31.25%	
	Best Solution.	2	-	1	-	-	3	18.75%	
	Average.	3	1	-	-	-	4	25.00%	
	Std Dev.	4	1	-	-	1	6	37.50%	
	Aggregate	12	2	2	-	2	**18**	**28.12%**	**2**
SAO	Conv Time (s).	2	-	-	1	-	3	18.75%	
	Best Solution.	3	-	-	-	-	3	18.75%	
	Average.	1	-	1	-	-	2	12.50%	
	Std Dev.	1	-	1	-	-	2	12.50%	
	Aggregate	7	-	2	1	-	**10**	**15.63%**	**3**

The performance results of the algorithms are detailed in Table 3 and summarized in Table 4 using ABC. Statistical figures showed that SASOS obtained the leading ABC of 56.25% followed by SOS and SAO with 28.12% and 15.63%, respectively. As a result, SASOS is ranked first, while SOS and SAO are ranked second and third, respectively. A convergence study is also carried out to evaluate and compare the fitness functions of the algorithms in the performance graphs.

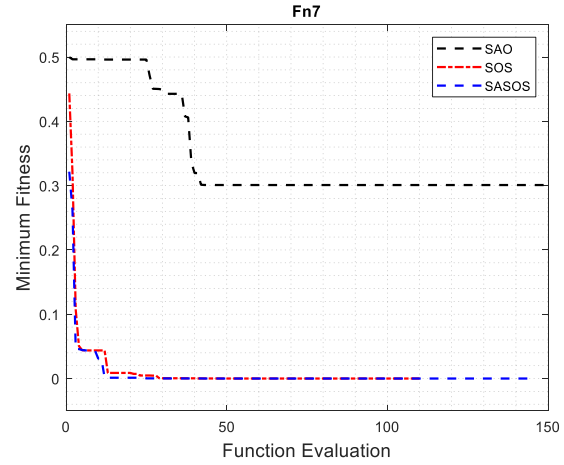
Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

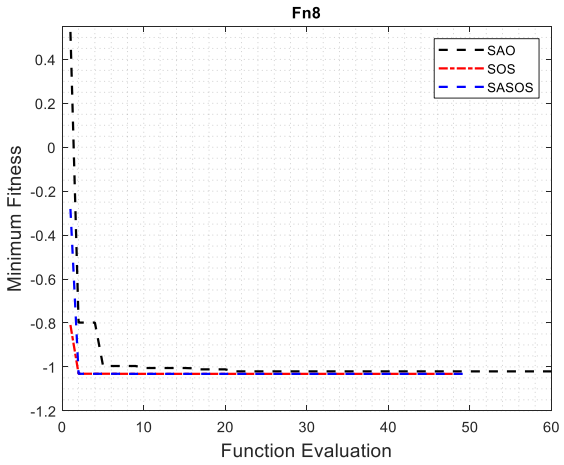
4.1 Convergence Study

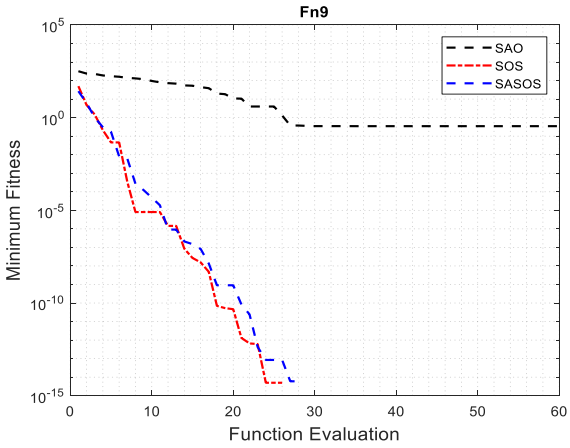
The convergence study is conducted on the CEC functions to evaluate the fitness values of each algorithm. The superimposed 1000 iterations are applied to analyze the performance of the algorithms. Figure 3 shows the convergence of the algorithms on each CEC function.

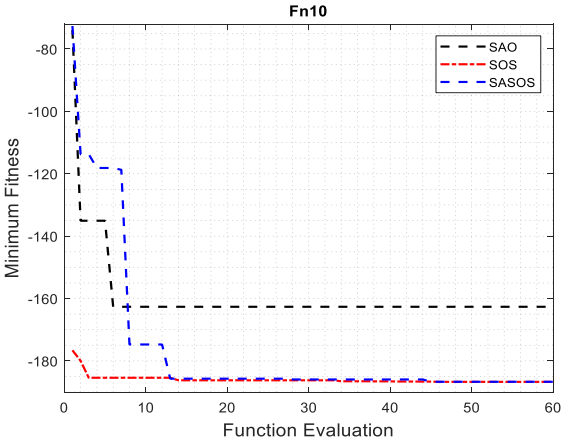


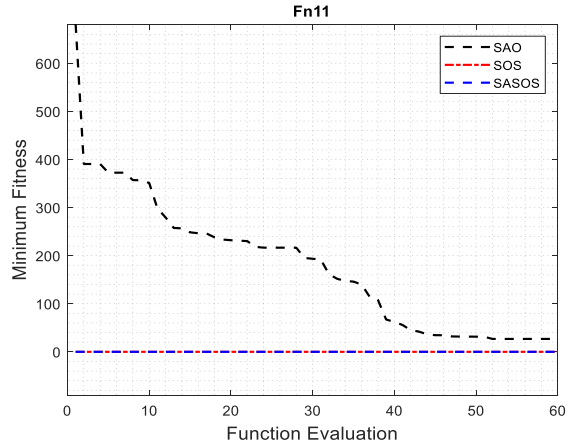
Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

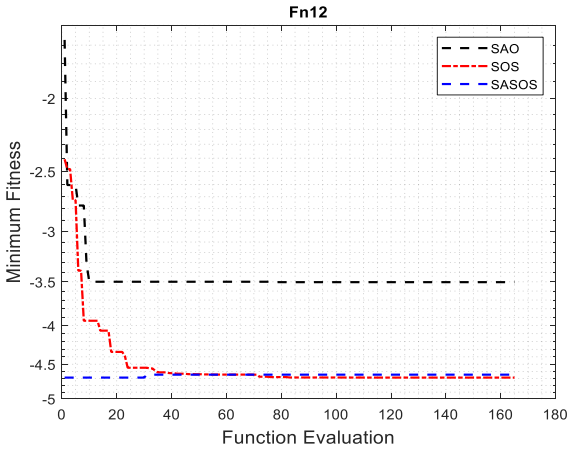












Sule Lamido University Journal of Science & Technology Vol. 6 No. 1&2 [March, 2023], pp. 44-58
https://doi.org/10.56471/slujst.v6i.350
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

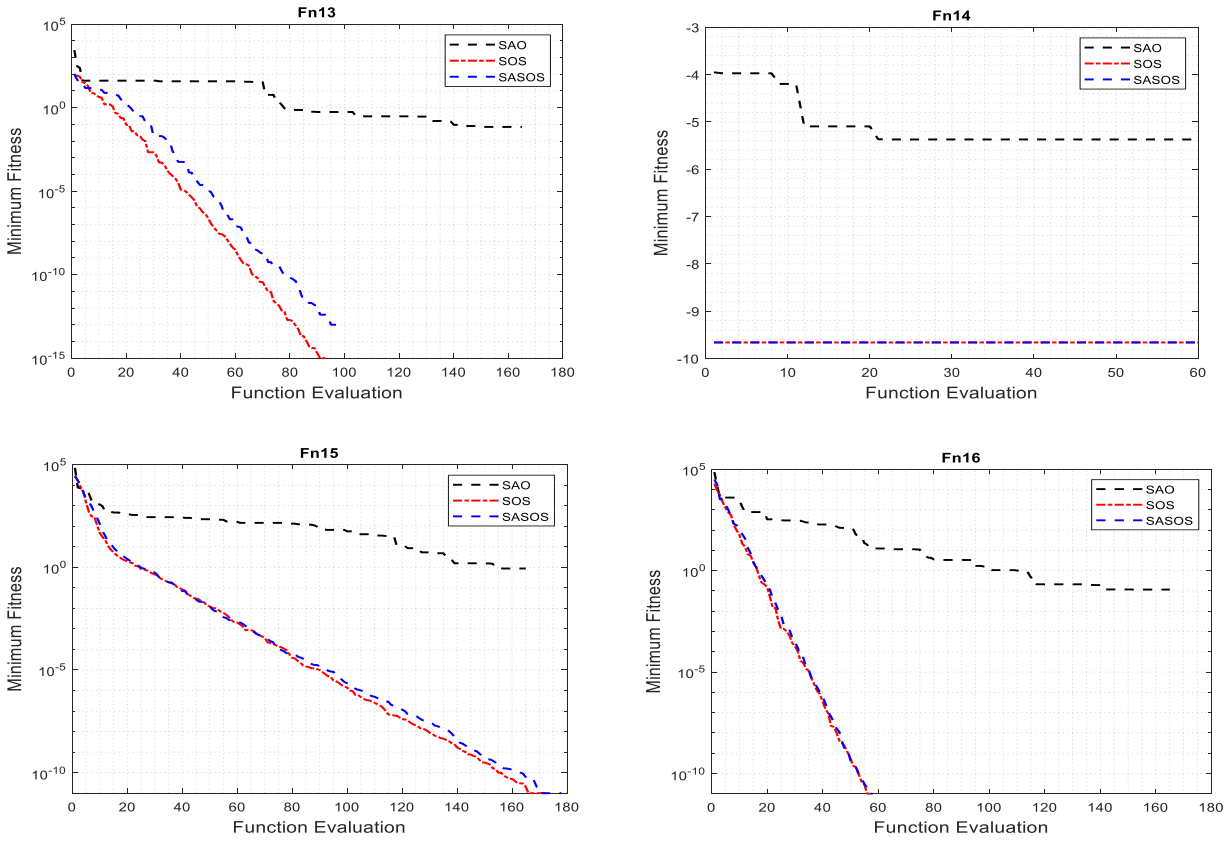


Figure 3. Convergence graphs of algorithms on multidimensional CEC functions

The convergence graphs that compare the three algorithms in 16 CEC functions are shown in Figure 3. From these graphs, it can be seen that SASOS obtained DCG in Fn1, Fn2, Fn7, Fn9, Fn11, Fn13, and Fn16, respectively. The SOS obtained DCG in Fn4, Fn10, and Fn12. Whereas SAO has DCG in Fn6 and Fn8, respectively. However, no DCG is obtained in Fn3, Fn5, Fn14 and Fn15, respectively, because their values exist in separable negative range of the global minimum. The counts showed that SASOS performed better in 12 functions with 58.33% DCG, followed by SOS and SAO with 25.00% and 16.67% DCG, respectively.

5. Conclusion

In this paper, a hybrid SASOS algorithm is developed for optimal performance balance between speed and accuracy in the global search space. The algorithm developed was evaluated in 16 CEC functions and compared with regular SOS and SAO algorithms using ABC and DCG. The simulation results revealed that SASOS obtained 56.25% ABC, followed by SOS and SAO with 28.12% and 15.63% ABC, respectively. In the convergence study, SASOS obtained 58.83% DCG in 12 functions, while SOS and SAO obtained 25.00% and 16.67% DCG, respectively. Based on these analyses, it can be seen that SASOS outperformed the SOS and SAO algorithms on the selected multidimensional CEC functions. This showed that SASOS balanced the diversification and intensification of the SAO and SOS algorithms. In subsequent work, SASOS is recommended for various applications of engineering optimization such as in power loss minimization, optimal placement of distributed generators, microgrid control optimization, and optimal control of stochastic load disturbances.

Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

References

Abdullahi, M., Ngadi, M. A., & Abdulhamid, S. M. (2016). Symbiotic Organism Search optimization based task scheduling in cloud computing environment. *Future Generation Computer Systems*, *56*, 640–650. https://doi.org/10.1016/j.future.2015.08.006

Abdullahi, M., Ngadi, M. A., Dishing, S. I., Abdulhamid, S. M., & Usman, M. J. (2020). A survey of symbiotic organisms search algorithms and applications. *Neural Computing and Applications*, *32*(2), 547–566. https://doi.org/10.1007/s00521-019-04170-4

Al-Tameemi, Z. H. A., Lie, T. T., Foo, G., & Blaabjerg, F. (2022). Optimal Coordinated Control of DC Microgrid Based on Hybrid PSO–GWO Algorithm. *Electricity*, *3*(3), 346–364. https://doi.org/10.3390/electricity3030019

Cheng, M. Y., & Prayogo, D. (2014). Symbiotic Organisms Search: A new metaheuristic optimization algorithm. *Computers and Structures*, *139*, 98–112. https://doi.org/10.1016/j.compstruc.2014.03.007

Goldanloo, M. J., & Gharehchopogh, F. S. (2022). A hybrid OBL-based firefly algorithm with symbiotic organisms search algorithm for solving continuous optimization problems. In *The Journal of Supercomputing* (Issue August 2021). Springer US. https://doi.org/10.1007/s11227-021-04015-9

Guo, J., Sun, Z., Tang, H., Jia, X., Wang, S., Yan, X., Ye, G., & Wu, G. (2016). Hybrid Optimization Algorithm of Particle Swarm Optimization and Cuckoo Search for Preventive Maintenance Period Optimization. *Discrete Dynamics in Nature and Society*, *2016*. https://doi.org/10.1155/2016/1516271

Hasanien, H. M., & El-Fergany, A. A. (2017). Symbiotic organisms search algorithm for automatic generation control of interconnected power systems including wind farms. *IET Generation, Transmission and Distribution*, *11*(7), 1692–1700. https://doi.org/10.1049/iet-gtd.2016.1245

Jafari, A., Khalili, T., Babaei, E., & Bidram, A. (2020). A Hybrid Optimization Technique Using Exchange Market and Genetic Algorithms. *IEEE Access*, *8*, 2417–2427. https://doi.org/10.1109/ACCESS.2019.2962153

Jia, H., Xing, Z., & Song, W. (2019). A New Hybrid Seagull Optimization Algorithm for Feature Selection. *IEEE Access*, *7*, 49614–49631. https://doi.org/10.1109/ACCESS.2019.2909945

Kang, L., Liu, D., Wu, Y., & Ping, G. (2023). Two-Stage Hybrid Optimization Algorithm for Silicon Single Crystal Batch Scheduling Problem under Fuzzy Processing Time. *Mathematical Problems in Engineering*, *2023*, 1–14. https://doi.org/10.1155/2023/3816574

Kesavan, V., Kamalakannan, R., Sudhakarapandian, R., & Sivakumar, P. (2019). Materials Today : Proceedings Heuristic and meta-heuristic algorithms for solving medium and large scale sized cellular manufacturing system NP-hard problems : A comprehensive review. *Materials Today: Proceedings*, *xxxx*. https://doi.org/10.1016/j.matpr.2019.05.363

Lang, C., & Jia, H. (2019). *Kapur ' s Entropy for Color Image Segmentation Based on a Hybrid Whale Optimization Algorithm*. https://doi.org/10.3390/e21030318

Mafarja, M. M., & Mirjalili, S. (2017). Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. *Neurocomputing*, *260*, 302–312. https://doi.org/10.1016/j.neucom.2017.04.053

Mohammed, S., Sha'aban, Y. A., Umoh, I. J., & Salawudeen, A. T. (2022). Optimal design of full order state observer for active surge control in centrifugal compressors using genetic algorithm. *Telkomnika (Telecommunication Computing Electronics and Control)*, *20*(2), 405–415. https://doi.org/10.12928/TELKOMNIKA.v20i2.17440

Mokarram, A., Jafar, M., Mokarram, M. J., & Niknam, T. (2019). *Hybrid optimization algorithm to solve the nonconvex multiarea economic dispatch problem Hybrid Optimization Algorithm to Solve the Nonconvex Multiarea Economic Dispatch Problem*. *13*, 3400–3409.

Pani, S. K. (2021). SSII : Secured and High-Quality Steganography Using Intelligent Hybrid Optimization Algorithms for IoT. *IEEE Access*, *9*, 87563–87578. https://doi.org/10.1109/ACCESS.2021.3089357

Radosavljević, J., Klimenta, D., Jevtić, M., & Arsić, N. (2015). *optimization and gravitational search algorithm Optimal Power Flow Using a Hybrid Optimization Algorithm of Particle Swarm Optimization and Gravitational Search Algorithm*. *October*. https://doi.org/10.1080/15325008.2015.1061620

Saha, S., & Mukherjee, V. (2018). A novel chaos-integrated symbiotic organisms search algorithm for global optimization. *Soft Computing*, *22*(11), 3797–3816. https://doi.org/10.1007/s00500-017-2597-4

Salawudeen, A. T., Mu'azu **, M. B., Sha'aban, Y. A., & Adedokun, E. A. (2019). on the Development of a Novel

Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

Smell Agent Optimization (Sao) for Optimization Problems. *I-Manager's Journal on Pattern Recognition*, *5*(4), 13. https://doi.org/10.26634/jpr.5.4.15677

Salawudeen, A. T., Mu'azu, M. B., Sha'aban, Y. A., & Adedokun, A. E. (2021). A Novel Smell Agent Optimization (SAO): An extensive CEC study and engineering application[Formula presented]. *Knowledge-Based Systems*, *232*, 107486. https://doi.org/10.1016/j.knosys.2021.107486

Sayed, E. A., Sameh, M. A., Attia, M. A., & Badr, A. O. (2023). Enhancement of PV performance by using hybrid TLBO-EO optimization. *Ain Shams Engineering Journal*, *14*(3), 101892. https://doi.org/10.1016/j.asej.2022.101892

Sen, T., & Mathur, H. D. (2016). A new approach to solve Economic Dispatch problem using a Hybrid ACO-ABC-HS optimization algorithm. *International Journal of Electrical Power and Energy Systems*, *78*, 735–744. https://doi.org/10.1016/j.ijepes.2015.11.121

Shankar, G., & Mukherjee, V. (2016). Load frequency control of an autonomous hybrid power system by quasi-oppositional harmony search algorithm. *International Journal of Electrical Power and Energy Systems*, *78*, 715–734. https://doi.org/10.1016/j.ijepes.2015.11.091

Singh, N., & Hachimi, H. (2018). A New Hybrid Whale Optimizer Algorithm with Mean Strategy of Grey Wolf Optimizer for Global Optimization. *Mathematical and Computational Applications*, *23*(1), 14. https://doi.org/10.3390/mca23010014

Tao, X., Zhang, L., & Wang, F. (2022). Droop Control Optimization Strategy for Parallel Inverters in a Microgrid Based on an Improved Population Division Fruit Fly Algorithm. *IEEE Access*, *10*, 24877–24894. https://doi.org/10.1109/ACCESS.2022.3145965

Tijani, S. A., Mu'azu, M., Sha'aban, Y., & Adedokun, E. A. (2018). *From Smell Phenomenon to Smell Agent Optimization (SAO): A Feasibility Study*. *Ci*. https://www.researchgate.net/publication/330728183

Vinaykumar, V. N., Babu, J. A., & Frnda, J. (2023). Optimal guidance whale optimization algorithm and hybrid deep learning networks for land use land cover classification. *EURASIP Journal on Advances in Signal Processing*, *2023*(1). https://doi.org/10.1186/s13634-023-00980-w

Yu, D., Wang, Y., Liu, H., & Jermsittiparsert, K. (2019). System identification of PEM fuel cells using an improved Elman neural network and a new hybrid optimization algorithm. *Energy Reports*, *5*, 1365–1374. https://doi.org/10.1016/j.egyr.2019.09.039

Zellagui, M., Lasmari, A., Settoul, S., El-Sehiemy, R. A., El-Bayeh, C. Z., & Chenni, R. (2021). Simultaneous allocation of photovoltaic DG and DSTATCOM for techno-economic and environmental benefits in electrical distribution systems at different loading conditions using novel hybrid optimization algorithms. *International Transactions on Electrical Energy Systems*, *31*(8). https://doi.org/10.1002/2050-7038.12992

Zhang, Y. J., Yan, Y. X., Zhao, J., & Gao, Z. M. (2022). AOAAO: The Hybrid Algorithm of Arithmetic Optimization Algorithm With Aquila Optimizer. *IEEE Access*, *10*, 10907–10933. https://doi.org/10.1109/ACCESS.2022.3144431

Zukhri, Z., & Paputungan, I. V. (2013). A Hybrid Optimization Algorithm based on Genetic Algorithm and Ant Colony Optimization. *International Journal of Artificial Intelligence & Applications*, *4*(5), 63–75. https://doi.org/10.5121/ijaia.2013.4505