Engineering and Applied Science Research
https://www.tci-thaijo.org/index.php/easr/index
Published by the Faculty of Engineering, Khon Kaen University, Thailand

Improved Symbiotic Organism Search (I-SOS) for global numerical optimization

Umar Umar*[1)], Faanzir Faanzir[1)], Iswan Iswan[1)], Muhammad Said[1)] and Haryati Haryati[2)]

[1)]Department of Electrical Engineering, Faculty of Engineering, Khairun University, Ternate, 97719, Indonesia
[2)]Department of Architecture, Faculty of Engineering, Khairun University, Ternate, 97719, Indonesia

Received 8 July 2023
Revised 10 September 2023
Accepted 13 September 2023

Abstract

The no free lunch theory states that no specific heuristic method can effectively solve all problems. This theory has created opportunities for researchers to improve existing heuristic methods or even discover new approaches. One algorithm that has gained considerable attention from researchers is the Symbiotic Organism Search (SOS) algorithm. Its appeal lies in its simplicity and minimal parameter requirements, making it applicable to various problem domains. However, the SOS algorithm also has its limitations. This study focuses on the enhancement of SOS by introducing a modified random weight (MRW) method during the parasitism phase, resulting in the Improved SOS (I-SOS) algorithm. The effectiveness of this algorithm is tested in solving unconstrained problems using 26 benchmark functions and compared to several existing heuristic methods in the literature. The simulation results show that I-SOS outperforms basic SOS as well as several other algorithms.

Keywords: Benchmark, Enhancement, Heuristic, Modified random weight, No free lunch

1. Introduction

The shortcomings of deterministic methods in solving complex problems have led to the advancement of heuristic methods. According to the "no free lunch" (NFL) theory, no individual heuristic approach can effectively address all optimization problems [1]. This theory has opened chances for researchers and engineers to enhance existing methods or even discover new heuristic approaches. Various heuristic methods have been successfully applied to solve real-world engineering problems. Genetic Algorithm (GA) has been used for optimizing distributed generation [2, 3]; Improved stochastic fractal search algorithm (ISFS) for automatic generation control (AGC) in power systems [4]; Particle Swarm Optimization (PSO) based on support vector machine (SVM) for structural optimization [5]; Grey Wolf Optimization (GWO) for wireless sensor network optimization [6], and Modified Salp Swarm Algorithm (mSSA) for optimization in practical engineering problems [7]. One such algorithm that has garnered considerable attention from engineers is Symbiotic Organism Search (SOS) algorithm [8]. Its appeal lies in its simplicity and minimal parameter requirements, which make it easily applicable to various problem domains [8].

While SOS is widely regarded as a robust algorithm [8-12], it is not without its limitations, as highlighted by several works. Al-Sharhan and Omran [12] demonstrated that the improved organisms process in the mutualism and commensalism phases can quickly impact the entire ecosystem, resulting in low organism variability and premature convergence. Additionally, the random selection of organisms can lead to incorrect selections, wherein organisms with good fitness may be chosen during the phase of parasitism, hindering the introduction of new individuals, and reducing ecosystem variability [13]. Another challenge with SOS algorithm is achieving a balance between exploration and exploitation [14-16]. This refers to the algorithm's ability to strike a suitable balance between searching for new solutions and exploiting already discovered solutions. Maintaining this balance is crucial for effective optimization. Furthermore, SOS has been criticized for its inefficient computational time [14, 16-18], indicating that it may require significant computational resources or time to converge towards an optimal solution.

There have been several studies conducted to improve the performance of the SOS algorithm. Al-Sharhan and Omran [12] proposed the use of ring propagation method to increase organism variability. Rodrigues et al. [13] introduced grouping, assignment, and ranking methods. Kumar et al. [14] focused on finding a balance between exploration and exploitation in SOS by introducing the adaptive benefit factor. Furthermore, Celik [15] suggested changes were made to the parasitism phase. The parasitism phase was divided into two groups with a 50% probability, and the parasitism vector was determined using quasi-opposite based learning (QOBL). Modifications to all phases were suggested in [16]. During the mutualism phase, each organism utilized the same benefit factor, while the random coefficient was limited to a certain range. The parasitism phase was excluded from the algorithm to simplify it. To enhance convergence speed and achieve a global optimum, the chaotic local search [18], and quasi-oppositional techniques were employed to improve optimization performance [19]. Additionally, adaptive Cauchy mutation methods were introduced in [20]. Modifications to the benefit factor and mutual vector are proposed by Chakraborty et al. [21], while the memory mechanism is proposed by Zhao and Liu [22]. The recombination and mutation methods are proposed by Yang et al. [23]. The adaptive chaotic local search method is

*Corresponding author.
Email address: umarmadjid94@gmail.com
doi: 10.14456/easr.2023.52

suggested by Dash et al. [24], while the clustering method is proposed by Yang and Sutrisno [25]. These diverse modifications were proposed to address specific limitations of the SOS algorithm and improve its overall effectiveness in optimizing complex problems.

According to previous research [13-17], the parasitism phase plays a significant role in enhancing the performance of the SOS algorithm. In this study, an improved version of SOS called I-SOS is introduced. The modification is implemented by creating a subphase within the parasitism phase and integrating the modified random weight (MRW) from the crow search algorithm (CSA). This modification aims to reduce exploration by limiting the number of organisms generated through the original parasitism. On the other hand, the use of MRW can increase the exploitation of organisms, thus enhancing the balance between exploration and exploitation.

2. Improved Symbiotic Organism Search (I-SOS)

Cheng and Prayogo [9] introduced the Symbiotic Organism Search (SOS) algorithm in 2014, which mimics the symbiotic relationships found in natural ecosystems. These relationships include mutualism, commensalism, and parasitism. The mutualism phase of the SOS algorithm symbolizes the beneficial interaction between organisms, where they mutually assist each other. This relationship can be formulated as follows [9]:

$$X_{iN} = X_i + rand\ (0,1) \times (X_{best} - MV \times bf_1) \tag{1}$$

$$X_{kN} = X_k + rand\ (0,1) \times (X_{best} - MV \times bf_2) \tag{2}$$

Where X_i and X_k are two interacting individuals. X_{iN} and X_{kN} are the new organisms produced from the interaction in the mutualism phase. Here, i and k are integers. X_{best}, *bf* and MV represent the best organism, benefit factor, and mutual vector respectively. MV and *bf* are formulated as follows [9]:

$$MV = 0.5 \times (X_i + X_k) \tag{3}$$

$$bf_1 = 1 + round\ (rand\ (0,1)) \tag{4}$$

$$bf_2 = 1 + round\ (rand\ (0,1)) \tag{5}$$

The symbiotic relationship in the commensalism phase results in new individuals as follows [3]:

$$X_{iN} = X_i + rand\ (-1,1) \times (X_{best} - X_k) \tag{6}$$

During the parasitism phase, a symbiotic relationship is observed where one organism benefits while the other experiences drawbacks. This relationship is illustrated by creating organisms as parasitic vectors and other organisms as hosts. In this scenario, a parasitic vector will persist and supplant its host if it exhibits superior fitness in comparison to the host's fitness. Conversely, the host will endure and replace the parasitic vector if it demonstrates better fitness than the parasitic vector.

Striking a harmony between exploitation and exploration of organisms is crucial factor for achieving the global optimum. In the parasitism phase, most new organisms with lower fitness values are eliminated from the population. The dominant organisms within the ecosystem are the new ones with higher fitness levels. As a result, the organism variability decreases, and the search space primarily focuses on organisms with high fitness. This characteristic can lead to premature convergence in the SOS algorithm. The absence of parameter tuning in the SOS algorithm makes it simple and user-friendly, but it also poses a problem on the other hand. The lack of parameter tuning that guides the solution search process leads to over-exploration during the parasitism phase, resulting in inefficient computation times [15].

To tackle this issue, the parasitism phase is split into two sub-phases: original parasitism and random weight parasitism. In the I-SOS algorithm, the original parasitism phase remains unchanged and follows the same approach as the basic SOS algorithm. However, the random weight parasitism introduces a modification by incorporating a modified random weight (MRW) from the crow search algorithm (CSA) as follow:

$$MRW = rand\ (0,1) \times rand\ (-2,\ 2) \tag{7}$$

The new organism in the parasitism phase is written as follows:

$$X_{iNew} = X_i + MRW \times (X_{best} - X_i) \tag{8}$$

Utilizing a split phase within the parasitism phase leads to organisms produced during this phase being equally sourced from both the original parasitism and MRW. As a result, the quantity of organisms generated through the original parasitism diminishes, effectively reducing exploration. Conversely, the utilization of MRW involving organism Xi in equation (8) has the potential to boost the SOS algorithm's exploitation capability. This outcome ensures the preservation of individual variability, while simultaneously enhancing the equilibrium between the abilities to exploit and explore. The flowchart of I-SOS is depicted in Figure 1. The value of α in the flowchart indicates the percentage of new organisms generated with random weights.

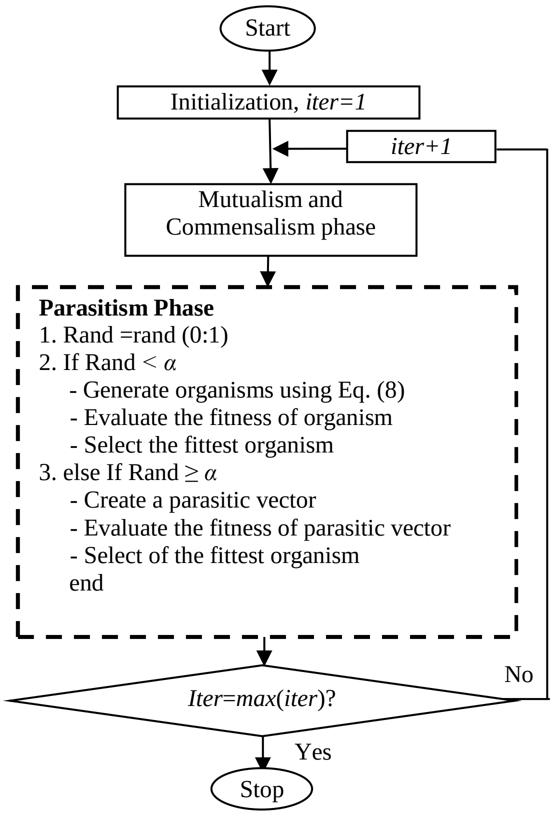


Figure 1 I-SOS flowchart

3. Results and discussions

3.1 Mathematical benchmark functions

The literature indicates a strong correlation between metaheuristic algorithms and numerical test problems. Benchmark test functions are mathematical functions that represent optimization problems [26]. These functions are optimized by searching for parameter values that yield the best solution. The problem space contains numerous sub-optimal solutions, characterized by hills and valleys, with the best solution hidden among them. The goal of optimization algorithms is to find the best solution as quickly as possible from a set of sub-optimal solutions in the search space. For validation, the I-SOS algorithm is used to solve unconstrained problems using 26 standard mathematical functions. The benchmark functions, dimensions (Dim), and minimum values (Min) are displayed in Table 1 [9].

Table 1 Benchmark functions

No	Function	Dim	Min	No	Function	Dim	Min
1	Beale	2	0	14	Zakharov	10	0
2	Easom	2	-1	15	Michalewicz 10	10	-9.6602
3	Matyas	2	0	16	Step	30	0
4	Boha chevsky 1	2	0	17	Sphere	30	0
5	Booth	2	0	18	Sum squares	30	0
6	Michalewicz 2	2	-1.8013	19	Quartic	30	0
7	Schaffer	2	0	20	Schwefel 22	30	0
8	Six Hump Camel Back	2	-1.03163	21	Schwefel 12	30	0
9	Boha chevsky 2	2	0	22	Rosenbrok	30	0
10	Boha chevsky 3	2	0	23	Dixon- Price	30	0
11	Schubert	2	-186.73	24	Rastrigin	30	0
12	Colville	4	0	25	Griewank	30	0
13	Michalewicz 5	5	-4.6877	26	Ackley 26	30	0

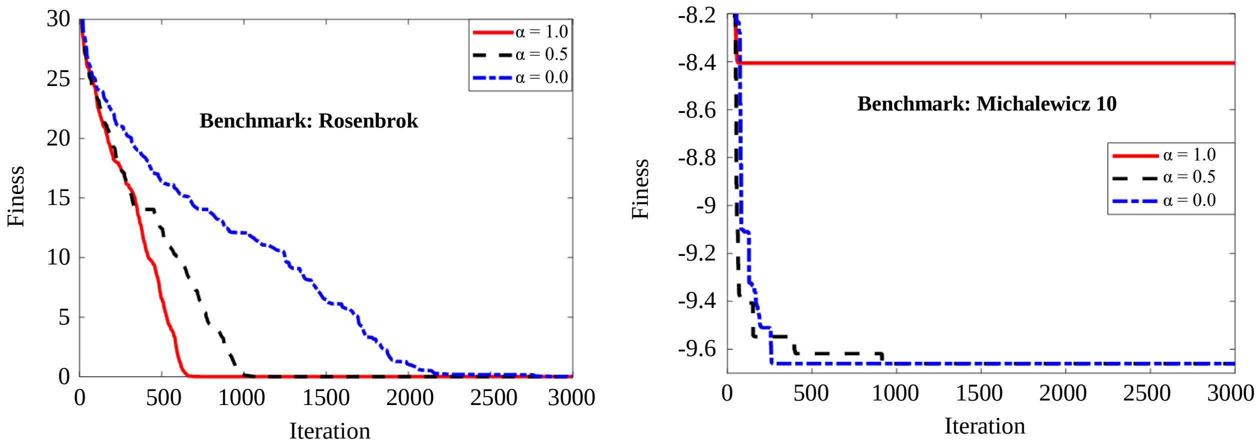
Table 2 Effect of random coefficient

Benchmark	Description	Random coefficient (α)						
		1.0	**0.9**	**0.7**	**0.5**	**0.3**	**0.1**	**0.0**
Michalewicz 10	Mean	-9.08060	-9.59689	-9.65541	-9.660152	-9.660152	-9.660152	-9.660152
	NFE_{avg}	9000.00	8085.00	5614.20	2900.10	2695.50	2538.90	2423.7
	time (s)	10.7991	9.5322	6.6471	3.4947	3.4394	3.2509	3.2193
	Running times	10	10	10	10	10	10	10
	Convergent times	0	4	8	10	10	10	10
Rosenbrok	Mean	9.28E-13	9.40E-13	9.709E-13	9.72E-13	8.933E-13	3.357E-11	0.266678
	NFE_{avg}	4153.5	4254.6	4743.3	5029.5	6433.8	8989.5	9000.00
	time (s)	4.182066	4.619358	5.20878	5.4980604	7.2807716	10.372944	10.47039
	Running times	10	10	10	10	10	10	10
	Convergent times	10	10	10	10	10	1	0

3.2 Effect of random coefficient (α)

To observe the influence of random coefficients (α) on the algorithm's performance, a simple experiment was conducted by varying the random coefficient from 1 to 0. A random coefficient of 1 means all organisms (100%) follow the parasitism phase through MRW, while a random coefficient of 0 means no organism goes through MRW (100% of organisms follow the original parasitism). The parameter settings used were ecosize=50 and maximum error=$1x10^{-12}$. The maximum number of iterations is 3000 or maximum number fitness evaluations (NFE)=9000. Mean, Time, and NFE average values were obtained after running the program 10 times. The simulation was performed on 2 benchmark functions, namely Michalewicz 10 and Rosenbrok. These two benchmarks were chosen because both functions have large dimensions, and in the tests conducted by Cheng and Prayogo [9], SOS could not reach the expected convergent point. The simulation results are shown in Table 2.

Table 2 shows that the convergence of the algorithm is highly influenced by the variation in the value of α. The simulation results using the Michalecz 10 function indicate that decreasing the value of α in the simulation improves the algorithm's performance. The NFE and convergence time decrease, while the number of convergences increases. However, different results are observed in the simulation using the Rosenbrok function. Decreasing the value of α actually worsens the algorithm's performance. The required time and NFE increase, while the number of convergences decreases. The simulation results for α=0.5 and α=0.3 show that the number of convergences for each benchmark function is 10 for 10 program runs (100%). The difference lies in the fact that the time and NFE for α=0.5 are slightly better compared to the value of α=0.3. The convergence curves of the two benchmark functions for α=1.0, α=0.5, and α=0.0 are shown in Figure 2. Figure 2 illustrates that the convergence curve for α=0.5 falls between the curves for α=0.0 and α=1.0. Therefore, this paper utilizes I-SOS with α=0.5, which means that 50% of new individuals are randomly weighted.



(a) Rosenbrok convergence with different α

(b) Michalewicz 10 convergence with different α

Figure 2 Resenbrok and Michalewicz 10 convergence with different α

3.3 Numerical optimization

The performance of I-SOS was evaluated using a set of 26 unconstrained mathematical functions as a benchmark [8]. The results obtained from the simulations were compared to those of Genetic Algorithm (GA), Differential Evolution (DE), Particle Swarm Optimization (PSO), Bees Algorithm (BA), Symbiotic Organism Search (SOS), and Quasi-Oppositional SOS (QOSOS). I-SOS was developed using Matlab and executed on a laptop equipped with a core 2 duo processor and 4 GB of RAM. The parameter configurations for GA, DE, PSO, and BA followed the guidelines provided in [9], while the settings for QOSOS were based on [19]. In the case of I-SOS and SOS, the parameters used were ecosize = 50; Maximum number of iterations = 3000 or maximum NFE=9000; Maximum error = $1x10^{-12}$. The mean, and average NFE (NFE_{avg}) values of I-SOS were computed based on 25 runs for every benchmark. Any values below $1x10^{-12}$ were considered equivalent to 0 [9]. Table 3 illustrates the performance of I-SOS compared to other algorithms in solving the 26 benchmark functions. The highlighted numbers indicate the best results for the respective benchmark.

Table 3 demonstrates that based on the convergence rate, I-SOS outperforms the GA, PSO, DE, BA, SOS, and QOSOS algorithms. I-SOS achieves a higher convergence rate by converging on 24 out of 26 benchmark functions tested, while SOS and QOSOS converge on 22 and 23 benchmark functions, respectively. GA exhibits the worst performance, converging on only 9 benchmark functions. I-SOS fails to converge only on the Quadratic and Dixon Price functions. However, in the case of the Quadratic function, although I-SOS does not converge to the expected value, it demonstrates a better mean value (5.894×10^{-5}) compared to SOS. This mean value is also quite close to the expected global minimum value of 0. As for the Dixon Price function, QOSOS is the only algorithm that can reach the global minimum value. Other algorithms, including SOS and I-SOS, only produce a minimum value of 0.66667.

Table 3 I-SOS performance compared to other algorithms.

No	Min	Description	GA [9]	PSO [9]	DE [9]	BA [9]	QOSOS [19]	SOS	I-SOS
1	**0.000**	Mean	**0.000**	**0.000**	**0.000**	1.88E-5	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	158,751	**86.283**
2	**-1.000**	Mean	**-1.000**	**-1.000**	**-1.000**	-0.99994	**-1.000**	**-1.000**	**-1.000**
		NFE_{avg}	NA	NA	NA	NA	NA	168,708	**102.345**
3	**0.000**	Mean	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	70,914	**49,308**
4	**0.000**	Mean	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	74.639	**56.31**
5	**0.000**	Mean	**0.000**	**0.000**	**0.000**	5.3E-4	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	264.828	**219.63**
6	**-1.8013**	Mean	**-1.8013**	-1.57287	**-1.8013**	**-1.8013**	**-1.8013**	**-1.8013**	**-1.8013**
		NFE_{avg}	NA	NA	NA	NA	NA	44.865	**32,004**
7	**0.000**	Mean	0.00424	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	299.751	**146.370**
8	**-1.03163**	Mean	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**
		NFE_{avg}	NA	NA	NA	NA	NA	143.631	**110.508**
9	**0.000**	Mean	0.06829	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	68.25	**50.163**
10	**0.000**	Mean	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	96.48	**66.366**
11	**-186.73**	Mean	**-186.73**	**-186.73**	**-186.73**	**-186.73**	**-186.73**	**-186.73**	**-186.73**
		NFE_{avg}	NA	NA	NA	NA	NA	432.945	**284.256**
12	**0.000**	Mean	0.01494	**0.000**	0.04091	1.11760	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	3977,274	**316.305**
13	**-4.6877**	Mean	-4.64483	-2.49087	-4.68348	**-4.6877**	-4.68348	**-4.6877**	**-4.6877**
		NFE_{avg}	NA	NA	NA	NA	NA	382.308	**277.065**
14	**0.000**	Mean	0.01336	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	226.671	**168.963**
15	**-9.6602**	Mean	-9.49683	-4.00718	-9.59115	**-9.6602**	-9.6598	-9.65982	**-9.6602**
		NFE_{avg}	NA	NA	NA	NA	NA	9000	**2930.628**
16	**0.000**	Mean	1.17E+03	**0.000**	**0.000**	5.12370	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	554.37	**501.363**
17	**0.000**	Mean	1.11E+03	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	180.228	**149.643**
18	**0.000**	Mean	1.48E+2	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	170.31	**142.080**
19	**0.000**	Mean	0.18070	0.00116	0.00136	**1.72E-6**	3.2708E-5	7.415E-05	5.894E-05
		NFE_{avg}	NA	NA	NA	NA	NA	9000	9000
20	**0.000**	Mean	11.0214	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	303.237	**253.517**
21	**0.000**	Mean	7.40E+3	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	191.871	**162.030**
22	**0.000**	Mean	1.96E+5	15.08862	18.20394	28.834	1.0354	0.2700	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	9000	**5475.108**
23	**0.000**	Mean	1.22E+3	0.6667	0.66667	0.6667	**0.000**	0.6667	0.6667
		NFE_{avg}	NA	NA	NA	NA	NA	9000	9000
24	**0.000**	Mean	52.92259	43.97714	11.71673	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	280.44	**198,117**
25	**0.000**	Mean	10.63346	0.01739	0.00148	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	186.12	**154.428**
26	**0.000**	Mean	14.67178	0.16462	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
		NFE_{avg}	NA	NA	NA	NA	NA	297.24	**249.006**
Global minimum			9	17	18	18	23	22	24

NA: Not available.

Regarding the convergence speed, I-SOS outperforms SOS on 24 benchmark functions, and in 2 other benchmark functions, both I-SOS and SOS do not converge. Based on the NFE, I-SOS is on average 32.29% faster than SOS. The comparison of convergence characteristics between I-SOS and SOS for solving the Schaffer function is illustrated in Figure 3.

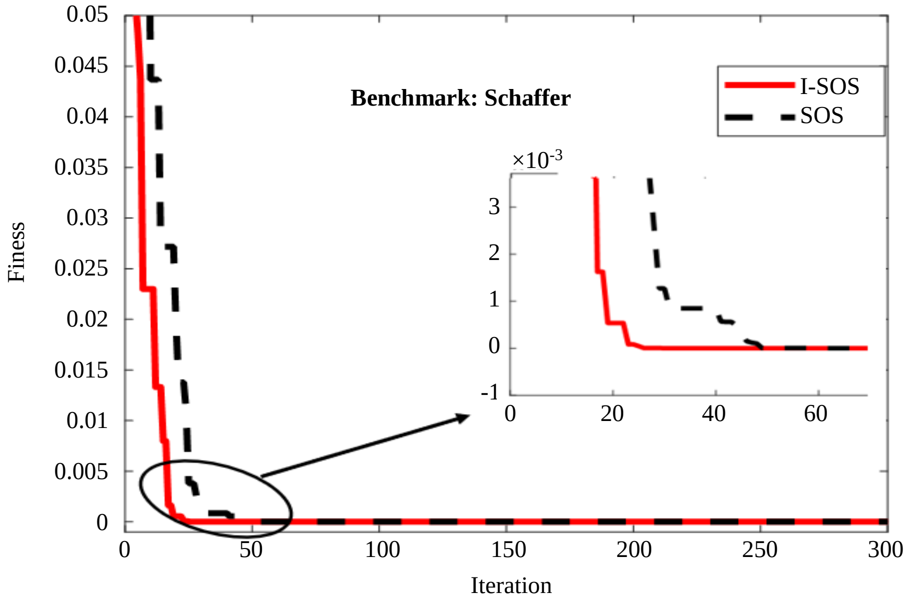


Figure 3 Convergence characteristics of I-SOS and SOS on the Schaffer function

4. Conclusions

This study suggests enhancing SOS by incorporating random weight during the parasitism phase. The inclusion of random weight in this phase aims to increase the diversity of organisms within the ecosystem, leading to the attainment of the optimal global value. The validation results using 26 benchmark functions demonstrate that I-SOS outperforms other methods in terms of NFE and convergence speed. I-SOS achieves convergence in 24 out of 26 benchmark functions with superior accuracy compared to alternative approaches. In terms of convergence speed, I-SOS excels across 24 benchmarks, being 32.29% faster than SOS. Based on the validation outcomes, it can be concluded that I-SOS exhibits a greater ability to discover optimal solutions.

5. References

[1] Wolpert DH, Macready WG. No free lunch theorems for optimization. IEEE Trans Evol Comput. 1997;1(1)67-82.

[2] Firdaus, Umar, Wibowo RS, Penangsang O. Optimal placement of intermittent DG renewable energy and capacitor bank for power losses reduction and voltage profile improvement in microgrids systems. J Eng Sci Technol. 2022;17(5):3539-54.

[3] Umar, Firdaus, Soeprijanto A, Penangsang O. Optimal expenditure and benefit cost based location, size and type of DGs in microgrids systems using adaptive real coded genetic algorithm. Telkomnika. 2018;16(1):10-7.

[4] Çelik E. Improved stochastic fractal search algorithm and modified cost function for automatic generation control of interconnected electric power systems. Eng Appl Artif Intell. 2020;88:1-20.

[5] Rosso MM, Cucuzza R, Trapani FD, Marano GC. Nonpenalty machine learning constraint handling using PSO-SVM for structural optimization. Adv Civ Eng. 2021;2021:1-17.

[6] Sert SA. A novel hybrid grey wolf optimization methodology for resource constrained networks. 30th signal processing and communications applications conference (SIU), 2022 May 15-18; Safranbolu, Turkey. USA: IEEE; 2022. p. 1-4.

[7] Çelik E, Öztürk N, Arya Y. Advancement of the search process of salp swarm algorithm for global optimization problems. Expert Syst Appl. 2021;182:1-16.

[8] Ezugwu AE, Prayogo D. Symbiotic organisms search algorithm: theory, recent advances, and applications. Expert Syst Appl. 2019;119:184-209.

[9] Cheng MY, Prayogo D. Organisms search: a new metaheuristic optimization algorithm. Comput Struct. 2014;139:98-112.

[10] Das B, Mukherjee V, Das D. DG placement in radial distribution network by symbiotic organisms search algorithm for real power loss minimization. Appl Soft Comput. 2016;49:920-36.

[11] Setyawan G, Umar, Soeprijanto A, Penangsang O. Symbiotic organism search for sizing and optimal location of distributed generation using novel sensitivity factor. IOP Conf Ser: Mater Sci Eng. 2019;588:1-7.

[12] Al-Sharhan S, Omran MGH. An enhanced symbiosis organisms search algorithm: an empirical study. Neural Comput Appl. 2016;29(11):1025-43.

[13] Rodrigues LR, Gomes JPP, Neto ARR, Souza AH. A modified symbiotic organisms search algorithm applied to flow shop scheduling problems. 2018 IEEE Congress on Evolutionary Computation (CEC); 2018 Jul 8-13; Rio de Janeiro, Brasil. USA: IEEE; 2018. p. 1-7.

[14] Kumar S, Tejani GG, Mirjalili S. Modified symbiotic organisms search for structural optimization. Eng Comput. 2019;35:1269-96.

[15] Celik E. A powerful variant of symbiotic organisms search algorithm for global optimization. Eng Appl Artif Intell. 2020;87:1-14.

[16] Do DTT, Lee J. A modified symbiotic organisms search algorithm for optimization of pin-jointed structures. Appl Soft Comput. 2017;61:683-99.

[17] Tsai HC. A corrected and improved symbiotic organisms search algorithm for continuous optimization. Expert Syst Appl. 2021;177:1-12.

[18] Saha S, Mukherjee V. A novel chaos-integrated symbiotic organisms search algorithm for global optimization. Soft Comput. 2018;22:3797-816.
[19] Nama S, Saha AK. An ensemble symbiosis organisms search algorithm and its application to real world problems. Decis Sci Lett. 2018;7(2):103-18.
[20] Miao F, Yao L, Zhao X. Symbiotic organisms search algorithm using random walk and adaptive Cauchy mutation on the feature selection of sleep staging. Expert Syst Appl. 2021;176:114887.
[21] Chakraborty S, Nama S, Saha AK. An improved symbiotic organisms search algorithm for higher dimensional optimization problems. Knowl-Based Syst. 2022;236:107779.
[22] Zhao P, Liu S. An improved symbiotic organisms search algorithm with good point set and memory mechanism. J Supercomput. 2023;79:11170-97.
[23] Yang K, Li C, Jing X, Wang Y, Huo Y, Ma H, et al. Energy optimization dispatching of islanded microgrid based on multi-agent system and improved symbiotic organisms search. IEEE 5th Conference on Energy Internet and Energy System Integration (EI2); 2021 Oct 22-24; Taiyuan, China. USa: IEEE; 2021. p. 1831-7.
[24] Dash SK, Mishra S, Raut U, Abdelaziz AY. An improved symbiotic organisms search algorithm for multi-objective simultaneous optimal allocation of DSTATCOM and DG units. 2021 International Conference in Advances in Power, Signal, and Information Technology (APSIT); 2021 Oct 8-10; Bhubaneswar, India. USA: IEEE; 2021. p. 1-6.
[25] Yang CL, Sutrisno H. A clustering-based symbiotic organisms search algorithm for high-dimensional optimization problems. Appl Soft Comput. 2020;97:1-37.
[26] Hussain K, Salleh MNM, Cheng S, Naseem R. Common benchmark functions for metaheuristic evaluation: a review. Int J Inform Visualization. 2017;1(4-2):218-23.