Assessing the Symbiotic Organism Search Variants using Standard Benchmark Functions

Nurul Asyikin Zainal¹, Fakhrud Din¹,² and Kamal Z. Zamli¹

¹ Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, 26600 Pekan, Malaysia
² Department of Computer Science & IT, University of Malakand, KPK, Pakistan

Email: kamalz@ump.edu.my

Abstract. Symbiotic Organism Search (SOS) is one of the latest meta-heuristic algorithms created to solve optimization problems. Combining the fact that this new algorithm is parameter-less (no need for tuning) and having a superior performance compared with other meta-heuristic algorithms, the interest to investigate and enhanced this algorithm had emerged. In this paper, we present a new version of SOS by looping the current algorithm rather than doing it one after the other. The target of this paper is to find the effect of changing the structure of algorithm from the original SOS using selected benchmark functions. Our findings indicate that using a loop structure can improve the solution in some of the benchmarks functions as compared to the original SOS.

1. Introduction
Optimizations are an important part of soft computing, and have been applied to many different fields such as geophysics [1], logistics [2], resources [3], sensor networks [4] and software testing [5-11]. Solving optimization problem often ensures effective use of the available resources. The goal of optimization is to find the optimal or near optimal solution from a large search space with minimum efforts (e.g. in terms of computational time and computer memory requirements).

Given that our world is finite, there is always a need to compensate between finding (near) optimal solution in the expense of resources and time. For these reasons, meta-heuristic based approaches have been proposed [12]. Meta-heuristic approaches are based on guided approach allowing balance controls of exploration (i.e. roams the random search space at the global scale) and exploitation (i.e. neighbourhood search by exploiting the current good solution) of the search space. These controls (i.e. implemented via algorithm parameters) are significant to prevent premature convergence and avoid being trapped in local optima.

In the last 30 years, many meta-heuristic algorithms have been proposed in the literature. Much of existing meta-heuristic algorithms take the analogy from nature, real world processes, as well as physical phenomenon. Examples of meta-Heuristics algorithms including Simulated Annealing (SA) [13], Teaching Learning based Optimization (TLBO) [14], and Particle Swarm Optimization algorithm (PSO) [15]. While much progress has been achieved as far as the development of new meta-heuristic algorithms, the No Free Lunch Theorem [16] dictates that there is no one size fits all approach indicating that no single algorithm can be good solutions for the whole class of optimization problems. For these reasons, research into new or enhanced meta-heuristic algorithm is still deemed necessary.
Inspired by the interactions of animals in the ecosystem for survival, a new meta-heuristics algorithm named Symbiotic Organism Search (SOS) [17] has been created to solve optimization problems. SOS controls the exploration and exploitation of the search space using three different stages namely Mutualism, Commensalism, and Parasitism. The fact that SOS is superior than other algorithms (although relatively new in the optimization realm) completely captures our interest to further investigate and enhance its capabilities.

In this paper, we modify the structure of its exploration and exploitation process. In particular, we will complete the exploration and exploitation in sequence rather than interleaving them one after the other as suggested by the original algorithm. These modifications aim to increase the effectiveness of exploration and sharing the information of the search space. We will compare these modifications with the original result of SOS using selected benchmark functions.

The rest of the paper is organized as follow; Section 2 covers the background of the Symbiotic Organism Search. Methodology for new implementations is covered is Section 3. Experimental study and the analysis of the results are covered in Section 4, followed by the concluding remarks.

2. Background study
Inspired by symbiosis relationship of organism to survive in an ecosystem, Cheng and Prayogo [17] create a nature based algorithm called Symbiotic Organism Search (SOS). In a nutshell, SOS algorithm manipulates three symbiosis interactions in order to calculate the position of its population (solutions) for exploration and exploitation in the search space. In this case, every organism is assumed to interact with other organism in all three symbiosis categories which are Mutualism, Commensalism, and Parasitism. The general SOS algorithm is shown in Figure 1:

![Pseudo Code of General SOS](image)

2.1. Mutualism Phase
In SOS, this mutualism phase is mimicking the interactions where two organisms get the benefits of living with each other. To increase the chances of survival in the ecosystem, Organism Xi is matched with a random Organism Xj and the candidate solutions are calculated by measuring the differences between the best solutions and the average of these two organisms (Mutual Vector). Therefore, interactions between two organisms which are far away will give a unique solution and enables that organism exploring a new search space. Equations 1 until 3 summarize the mutualism interaction.

\[
\text{Mutual Vector} = \frac{X_i + X_j}{2} \tag{1}
\]

\[
X_{\text{new}} = X_i + \text{rand}(0,1) \times (X_{\text{best}} - \text{Mutual Vector} \times BF_1) \tag{2}
\]

Where \(X_{\text{new}}\) = New value for variable \(X_i\); \(X_{\text{best}}\) = Current best solution; \(BF_1\) = random number either 1 or 2.

\[
X_{\text{new}} = X_j + \text{rand}(0,1) \times (X_{\text{best}} - \text{Mutual Vector} \times BF_2) \tag{3}
\]

Where \(X_{\text{new}}\) = New value for variable \(X_j\); \(X_{\text{best}}\) = Current best solution; \(BF_2\) = random number either 1 or 2.
2.2. **Commensalism Phase**

In Commensalism, only one organism gets the benefits from the interactions while the other organism might be unaffected by the collaborations. Based on this interactions, commensalism phase in SOS are calculated based on Equation 4. $X_j$ will be picked randomly and assumed to be a passive receiver in a commensalism interaction with organism $X_i$.

\[
X_{\text{new}} = X_i + r\cdot(1,1)(X_{\text{best}} - X_j)
\]  

(4)

Where $X_{\text{new}}$ = New value for variable $X_i$, $X_{\text{best}}$ = Current best solution.

In the calculation, $(X_{\text{best}} - X_j)$ is reflecting the benefits of $X_j$ towards the improvement of $X_i$ in searching for better candidate solutions. Best solutions are used as reference points so that the new candidate solution will be exploited around that region only. This will help to make convergence of solution faster.

2.3. **Parasitism Phase**

Parasitism involves survival of the fittest. Parasitism phase in SOS firstly duplicates organism $X_i$, modifying its characteristics by using some random number. This duplicated organism will be compared with randomly selected organism $X_j$. If $X_j$ has better fitness then $X_j$ is immune from the parasite (and the parasite is discarded). Meanwhile, if the parasite wins, the current $X_j$ will be killed and the parasite assumes the position of $X_j$ as highlighted in Equation 5.

\[
X_{\text{Parasite Vector}} = r\cdot(\text{lb},\text{ub})
\]

(5)

Where $X_{\text{Parasite Vector}}$ = New modified parasites $X_i$; $\text{lb}$ = Lower Bound; $\text{ub}$ = Upper Bound.

For all the phases, each organism interacts with other organism randomly. Every organism will go through each phase in sequence for every iteration until termination criteria are met.

3. **Methodology**

In the fundamental works of SOS, every organism will go through each phase one after the other in every iterations. This sequence might backfire as the organism might not be fully utilized to explore the search space as it was forced to converge prematurely without chances interacting with other organism. Therefore, this paper presents the implementation of Looping SOS (LoopSOS) where all organisms are given chance to explore the search space first before exploiting them. Modifications are done to the current SOS algorithm by introducing loop into the algorithm in each phase. Thus, the organisms are forced to undergo each phase first before allowing them to go for the next phase. The new LoopSOS algorithm is shown in Figure 2.

To validate the performance of LoopSOS, a set of benchmark functions has been adopted and its performance is compared against the original SOS algorithm. Table 1 gives the details of the benchmark functions based on the work of Cheng and Prayogo [17]. In this case, the execution will be stopped after 500,000 number of functions evaluation. Any values lower than 1E-12 will be treated and defined as 0. Mean and Standard Deviation of 30 consecutive running of algorithm are used as the evaluation matrices.
Population Initialization
REPEAT
  REPEAT
    Mutualism phase
    UNTIL (end of population)
  REPEAT
    Commensalism phase
    UNTIL (end of population)
  REPEAT
    Parasitism phase
    UNTIL (end of population)
UNTIL (termination criterion is met)

\[ \text{Figure 2. Pseudo Code of LoopSOS Algorithm.} \]

\begin{tabular}{|l|l|l|}
\hline
Function & D & Formulation & Search Range \\
\hline
Michalewicz 2 & 2 & \( f(x) = -\sum_{i=1}^{d} \sin(x_i) \left( \sin^{20} \left( \frac{ix_i}{\pi} \right) \right) \) & \([0, \pi]\) \\
Michalewicz 5 & 5 & \( f(x) = -\sum_{i=1}^{d} \sin(x_i) \left( \sin^{20} \left( \frac{ix_i}{\pi} \right) \right) \) & \([0, \pi]\) \\
Michalewicz 10 & 10 & \( f(x) = -\sum_{i=1}^{d} \sin(x_i) \left( \sin^{20} \left( \frac{ix_i}{\pi} \right) \right) \) & \([0, \pi]\) \\
Quartic & 30 & \( f(x) = \sum_{i=1}^{d} \left[ x_i^4 + \text{Rand}(0,1) \right] \) & 0 \\
Rosenbrock & 30 & \( f(x) = \sum_{i=1}^{d-1} \left[ 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right] \) & 0 \\
Booth & 2 & \( f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2 \) & \([-10, 10]\) \\
Rastrigin & 30 & \( f(x) = 10d + \sum_{i=1}^{d} x_i^2 - 10 \cos(2\pi x_i) \) & 0 \\
Sphere & 30 & \( f(x) = \sum_{i=1}^{d} x_i^2 \) & \([-100, 100]\) \\
SumSquare & 30 & \( f(x) = \sum_{i=1}^{d} x_i^2 \) & \([-10, 10]\) \\
Ackley & 30 & \( f(x) = -a \exp \left( -b \sqrt{\frac{1}{d} \sum_{i=1}^{d} x_i^2} \right) - \exp \left( \frac{1}{d} \sum_{i=1}^{d} \cos(cx_i) \right) + a + \exp(1) \) & \([-32, 32]\) \\
Griewank & 30 & \( f(x) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos \left( \frac{x_i}{\sqrt{i}} \right) + 1 \) & \([-600, 600]\) \\
\hline
\end{tabular}
4. Result and Analysis
The result obtained by LoopSOS against the original SOS is shown in Table 2 (i.e. in terms of the mean of optimal result and standard deviation after 30 consecutive runs).

Referring to Table 2, it is clear that the original SOS algorithm is also capable of finding the optimal. However, we can see improvement of the results from LoopSOS in comparison with the original SOS.

5. Discussions and Conclusions
We have modified the structure of original SOS algorithm so that all the three phases can be executed in total sequence. Our current results are encouraging as LoopSOS gives competitive results on a number of benchmark functions. More experiments need to be done in order to find the loopholes in the current SOS algorithm so that we can enhance the capability of this algorithm further.

| Function    | Min    | SOS     | LoopSOS |
|-------------|--------|---------|---------|
| Michalewicz 2 | Mean   | -1.8013 | -1.8013 |
|              | StdDev | 0       | 0       |
| Michalewicz 5 | Mean   | -4.6877 | -4.6877 |
|              | StdDev | 0       | 0       |
| Michalewicz 10 | Mean  | -9.6602 | -9.6598 |
|               | StdDev | 0.0013  | 0       |
| Quartic      | Mean   | 9.1E-5  | 0       |
|              | StdDev | 3.71E-5 | 0       |
| Rosenbrock   | Mean   | 1.04E-7 | 1.6031  |
|              | StdDev | 2.95E-7 | 1.4165  |
| Booth        | Mean   | 0.1287  | 0       |
|              | StdDev | 0.03382 | 0       |
| Rastrigin    | Mean   | 0       | 0       |
|              | StdDev | 0       | 0       |
| Sphere       | Mean   | 0       | 0       |
|              | StdDev | 0       | 0       |
| Sumsquare    | Mean   | 0       | 0       |
|              | StdDev | 0       | 0       |
| Ackley       | Mean   | 0       | 0       |
|              | StdDev | 0       | 0       |
| Griewank     | Mean   | 0       | 0       |
|              | StdDev | 0       | 0       |

Acknowledgements
This work is partially funded by the FRGS Grant from the Ministry of Higher Education Malaysia titled: A Reinforcement Learning Sine Cosine based Strategy for Combinatorial Test Suite Generation (grant no: RDU170103) and the UMP Matching Grants titled: Optimization using Artificial Bee Colony Algorithm (grant no: RDU182301).
References

[1] Godio A. A. and Santilano A. 2018 On the optimization of electromagnetic geophysical data: Application of the PSO algorithm. Journal of Applied Geophysics, 148: 163–174.

[2] Sakakibara, K. 2013 Adaptive optimization by dynamic programming heuristics for logistics planning in dynamic environments. Proc. Int. Symp. on System Integration (IEEE)

[3] Chaitanya, A.V.K., Rohit J., and Swathika, D.O.V.G. 2017 Optimum coordination of overcurrent relays in distribution systems using differential evolution and dual simplex methods. Proc. Int. Conf. on Computing Methodologies and Communication (IEEE)

[4] Parija, S. and Sahu P.J. 2017 A metaheuristic bat inspired technique for cellular network optimization, Proc. Int. Conf. on Man and Machine Interfacing (IEEE)

[5] Zamli K. Z., Din F, Baharom S. and Ahmed B. S. 2017 Fuzzy adaptive teaching learning-based optimization strategy for the problem of generating mixed strength t-way test suites. Engineering Applications of Artificial Intelligence, 59: 35-50

[6] Zamli K. Z., Din F., Ahmed B. S. and Bures M. 2018 A hybrid q-learning sine-cosine-based strategy for addressing the combinatorial test suite minimization problem. PLOS ONE, 13 (5): 1-18

[7] Younis M. I., Zamli K.Z., and Mat Isa N.A. 2008 A strategy for grid based t-way test data generation, Proc. Int. Conf. on Distributed Framework and Application (IEEE)

[8] Ahmed B.S, Gambardella L.M., and Afzal W. and Zamli K.Z. 2017 Handling constraints in combinatorial interaction testing in the presence of multi objective particle swarm and multitreading. Information Software Technology, 86:20-36

[9] Nasser A B, Zamli K Z, Alsewari A A and Ahmed B S 2018 Hybrid flower pollination algorithm strategies for t-way test suite generation PLOS ONE 13

[10] Younis M I, Zamli K Z and Isa N A M 2008 Algebraic strategy to generate pairwise test set for prime number parameters and variables. In: 2008 International Symposium on Information Technology, (Kuala Lumpur: IEEE) pp 1-4

[11] Zamli K Z, Zamli N, Ashidi M, Isa N A M, Fadel J, Klaib S N and Azizan S N 2007 A tool for automated test data generation (and execution) based on combinatorial approach International Journal of Software Engineering and its Applications 1

[12] Glover, F. and G.A. Kochenberger 2003 Handbook of metaheuristics (Kluwer Academic Publishers)

[13] Kirkpatrick, S., Gelatt C.D., and Vecchi M.P. 1983 Optimization by simulated annealing. Science, 220 (4589): 671–680.

[14] R.V.Rao, Savsani V.J., and Vakharia D.P. 2011 Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design, 43 (3): 303-315.

[15] Kennedy, J. and Eberhart, R. 1995 Particle swarm optimization. Proc. Int. Conf. on Neural Networks (IEEE)

[16] Wolpert, D.H. and Macready W.G. 1997 No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1997. 1 (1): 67–82

[17] Cheng, M.-Y. and Prayogo, D. 2014 Symbiotic organisms search: A new metaheuristic optimization algorithm. Computers and Structures, 139: 98–112.