Civil Engineering Dimension, Vol. 19, No. 1, March 2017, 36-43
ISSN 1410-9530 print / ISSN 1979-570X online

DOI: 10.9744/CED.19.1.36-43

A Novel Implementation of Nature-inspired Optimization for Civil Engineering: A Comparative Study of Symbiotic Organisms Search

Prayogo, D.[1]*, Cheng, M.Y.[2], and Prayogo, H.[1]

Abstract: The increasing numbers of design variables and constraints have made many civil engineering problems significantly more complex and difficult for engineers to resolve in a timely manner. Various optimization models have been developed to address this problem. The present paper introduces Symbiotic Organisms Search (SOS), a new nature-inspired algorithm for solving civil engineering problems. SOS simulates mutualism, commensalism, and parasitism, which are the symbiotic interaction mechanisms that organisms often adopt for survival in the ecosystem. The proposed algorithm is compared with other algorithms recently developed with regard to their respective effectiveness in solving benchmark problems and three civil engineering problems. Simulation results demonstrate that the proposed SOS algorithm is significantly more effective and efficient than the other algorithms tested. The proposed model is a promising tool for assisting civil engineers to make decisions to minimize the expenditure of material and financial resources.

Keywords: Constrained optimization; nature-inspired; symbiotic organisms search; symbiotic relationship.

Introduction

In recent decades, design optimization has become a critical and challenging activity that has gained in importance in the field of civil engineering. A goal of designers is to obtain optimal solutions in order to reduce construction project costs. Optimization allows designers to create better designs that reduce expenditures of material and financial resources as well as time. However, modern engineering design problems have increased tremendously in complexity and now frequently address complicated objective functions with large numbers of design variables and constraints [1]. This complexity has inspired numerous studies worldwide with the shared goal of developing a model that effectively optimizes current civil engineering problems.

Many optimization methods have been introduced over the past four decades. Gradient-based methods were the first of these methods to be widely used in solving decision-making problems in civil engineering [2]. These methods are often inadequate in dealing with the complexities inherent in many of today's optimization problems due to poor handling of large-scale variables and constraints. Additionally, these methods also use analyses that require gradient information to improve initial solutions. However, the designers usually have insufficient knowledge to locate the initial solutions, as they have no way to identify the most promising area for the global optimum of the current problem. Therefore, these gradient-based search methods frequently fail to converge on global optimum because of failed guesswork in defining the area of the global optimum. The above concerns have encouraged researchers to work to develop better optimization models.

The field of nature-inspired algorithms has been studied extensively with regard to its potential to solve optimization problems due to its superior performance in handling models that are highly nonlinear and complex. One of the most significant advantages of nature-inspired algorithms is that these algorithms do not use gradients to explore and exploit the problem search space. Instead, they combine natural pattern rules and randomness to identify near-optimum solutions efficiently [3]. Examples of nature-inspired algorithms include: Genetic Algorithm (GA) [4], Particle Swarm Optimization (PSO) [5], Differential Evolution (DE) [6], and Artificial Bee Colony (ABC) [7].

In recent years, numerous studies have proposed nature-inspired approaches to solve civil engineering

[1] Department of Civil Engineering, Petra Christian University, Jalan Siwalankerto 121-131, Surabaya 60236, INDONESIA
[2] Department of Civil and Construction Engineering, National Taiwan University of Science and Technology, TAIWAN
*Corresponding author; e-mail: prayogo@petra.ac.id

Note: Discussion is expected before June, 1st 2017, and will be published in the "Civil Engineering Dimension", volume 19, number 2, September 2017.

Received 19 January 2017; revised 18 February 2017; accepted 04 March 2017.

problems. In construction management, nature-inspired algorithms have been used to solve problems such as project site layout [8], time-cost trade-off [9], and resource leveling [10]. In structural engineering, examples of nature-inspired applications include: truss design [11,12] and frame design [13]. Nature-inspired algorithms have also been used in dealing with geotechnical problems [14], pavement engineering [15], and concrete mix design [16,17]. As civil engineering problems become more complex, new nature-inspired algorithms will continue to emerge.

A new nature-inspired algorithm called Symbiotic Organisms Search (SOS) has been developed by Cheng and Prayogo [18]. The SOS algorithm mimics the interactive behavior between living organisms in ecosystem. In the previous study, the performance of SOS has been compared with other nature-inspired techniques in numerous mathematical test functions and engineering problems. The comparison results indicate that SOS was able to achieve a better performance in terms of effectiveness and efficiency [18]. As a new nature-inspired algorithm, it is worthwhile to explore and investigate the SOS algorithm in seeking the global solution. This paper studies the effectiveness of Symbiotic Organisms Search (SOS) in solving various civil engineering optimization. SOS is first validated on benchmark functions and then tested on three practical civil engineering problems. The obtained results are then compared with well-known optimization techniques.

The Symbiotic Organisms Search (SOS)

Symbiotic Organisms Search (SOS) Algorithm

SOS is a new nature-inspired algorithm inspired by the natural phenomena of symbiotic interactions proposed by Cheng and Prayogo [18]. Over the past years, SOS has been proven to successfully solve various problems in different fields of research [19-22]. In surviving environmental change, the living organisms often develop symbiotic interactions among themselves. The most common examples of symbiotic interactions found in nature may be divided into three main categories:

1. Mutualism: This category describes the interactive behavior between two different living organisms that gain advantage mutually from that interaction. An example of mutualism is the relationships between oxpecker and zebra. Oxpecker lands on zebra, eating all the parasites. This activity benefits both zebra and oxpecker, since oxpecker collects foods and zebra gains pest control. Another example of mutualism is the relationship between bee and flower.
2. Commensalism: This category describes the interactive behavior between two different living organisms in which one gains advantage and the other is unaffected or neutral. An example of commensalism is the relationships between remora and shark. The remora gains an advantage by attaching itself to the shark and eats food leftovers. The shark is unaffected by remora fish activities and gains no benefit from the relationship. Another example of commensalism is the relationship between orchid and tree.
3. Parasitism: This category describes the interactive behavior between two different living organisms in which one gains advantage and the other is harmed. The anopheles mosquito transmits the plasmodium parasite into the human host. The parasite, thus, reproduces inside the body resulting the human host suffers malaria. Other examples of parasitism is the relationship between cuckoo and reed warbler.

In SOS algorithm, three phases of the search are performed mimicking the three symbiotic interactions namely mutualism, commensalism, and parasitism phase. By performing these three phases, SOS attempts to move a population (ecosystem) of possible solutions to a better region in the search space during the searching process for the optimal solution. In SOS, each solution in the population is known as an organism. Every organism is associated with its fitness value, which represents the survival advantage within the current environment. Through successive iterations, the fitness values of the organisms are improved by simulating the symbiotic interactions. The process of generating solutions through three phases is repeated until stopping criteria are satisfied. The source code for a MATLAB implementation of SOS is publicly available at http://140.118.5.71/sos/.

The next section provides further details on the three phases.

Mutualism Phase

The mutualism phase simulates the mutualism between two living organisms, eco_i and eco_j. The mechanism of mutualism is modeled in Equations (1) - (5).

$$eco_{mutual} = \frac{eco_i + eco_j}{2} \tag{1}$$

$$BF_1 = 1 + round\ (rand\ (0,1)) \tag{2}$$

$$BF_2 = 1 + round\ (rand\ (0,1)) \tag{3}$$

$$eco_{i\ new} = eco_i + rand\ (0,1) * (eco_{best} - eco_{mutual} * BF_1) \tag{4}$$

$$eco_{j\ new} = eco_j + rand\ (0,1) * (eco_{best} - eco_{mutual} * BF_2) \tag{5}$$

where eco_i is the i-th organism of the ecosystem, eco_j is the j-th organism of the ecosystem where $j \neq i$, BF_1 is the benefit factor matched to eco_i, BF_2 is the benefit factor matched to eco_j, eco_{best} represents the best organism in the current iteration, eco_{mutual} represents the relationship characteristic between organism eco_i and eco_j, $eco_{i\ new}$ and $eco_{j\ new}$ represent candidate solutions for eco_i and eco_j after their mutualistic interaction, respectively.

eco_{best} is the target point for every organism to increase its fitness during its interaction with another organism. Organisms eco_i and eco_j are updated only if their new ($eco_{i\ new}$ and $eco_{j\ new}$) fitness is better than their old fitness (eco_i and eco_j).

Commensalism Phase

The commensalism phase simulates the commensalism between two living organisms, eco_i and eco_j with eco_i gains advantage and eco_j is unaffected. The mechanism of commensalism is modeled in Equation (6).

$$eco_{i\ new} = eco_i + rand(-1,1) * (eco_{best} - eco_j) \qquad (6)$$

where eco_i is the i-th organism of the ecosystem, eco_j is the j-th organism of the ecosystem where $j \neq i$, $eco_{i\ new}$ represents candidate solutions for eco_i after their interaction, respectively.

Organism eco_i is updated only if its new fitness is better than its old fitness.

Parasitism Phase

The parasitism phase simulates the parasitism between two living organisms, eco_i and eco_j with eco_i gains advantage and eco_j is harmed. Organism eco_i serves a role similar to the anopheles mosquito and, thus, create an artificial parasite called $eco_{parasite}$.

Generally speaking, $eco_{parasite}$ is a clone of organism eco_i. To differentiate the $eco_{parasite}$ from eco_i, some random decision variables from the initial $eco_{parasite}$ will be modified randomly. The location of the modified decision variables is determined randomly using a random method. For each dimension, a uniform random number is generated. If the random number is less than 0.5, the variable will be modified by a random value generated by uniform distribution; otherwise, it will stay the same.

Organism eco_j serves as a host to the $eco_{parasite}$. If $eco_{parasite}$ has a better fitness value, it kills organism eco_j and replaces its position in the ecosystem. If the fitness value of eco_j is better, eco_j survives and the $eco_{parasite}$ can no longer exist in the ecosystem.

The Framework of the SOS Algorithm for the Design Optimization in Civil Engineering

Design objectives in design problems also have various other constraints including deflection, stress, material dimensions, pressure, and temperature. Many civil engineering problems may be expressed as constrained optimization problems. This paper handles the constraints using Deb's feasibility rules [23]. The use of SOS in constrained optimization problems that incorporate Deb's rules is summarized as follows.

Initialize Ecosystem

The SOS establishes an initial ecosystem by generating a matrix that contains uniform random numbers that exist within the given boundaries. After the initialization is complete, the initial best solution is calculated. The ecosystem is expressed as follows:

$$eco = \begin{bmatrix} eco_1 \\ \vdots \\ eco_{ecosize} \end{bmatrix} = \begin{bmatrix} x_{1.1} & \cdots & x_{1.D} \\ \vdots & \ddots & \vdots \\ x_{ecosize,1} & \cdots & x_{ecosize,D} \end{bmatrix}$$

In this step, the initial eco_{best} is determined by choosing the fittest organism in the initial ecosystem.

Simulate Interaction between Organisms through the Mutualism Phase

After the ecosystem initialization, each organism in the ecosystem will go through three phases, mutualism, commensalism, and parasitism. In the mutualism phase, eco_j is picked randomly from the ecosystem that is designated to interact with eco_i where i is start from 1, 2, 3, ... to $ecosize$, j is a random number which $\neq i$. New candidate solutions $eco_{i\ new}$ and $eco_{j\ new}$ are calculated using Equations (2) and (3), in which eco_{mutual} is determined using Equation (1) and Benefit Factors (BF_1 and BF_2) are determined using Equations (4) and (5). New candidate solutions $eco_{i\ new}$ and $eco_{j\ new}$ are compared to the old eco_i and eco_j. Deb's rules are implemented to retain the fittest solutions in the search space for the next iteration.

Simulate Interaction between Organisms through the Commensalism Phase

In the commensalism phase, another organism, eco_j, is picked randomly from the ecosystem to interact with eco_i. The new candidate solution $eco_{i\ new}$ is calculated using Equation (6) and compared to the older eco_i. Deb's feasibility rules are applied to identify the fittest organism as the solution to be carried forward into the next iteration.

Simulate Interaction between Organisms through the Parasitism Phase

In the parasitism phase, another organism, *eco_j*, is picked randomly from the ecosystem to be a host organism. *eco_{parasite}* is created by mutating the parent organism *eco_i* in random dimensions using distributed random numbers that are limited within a specific range. Deb's rules are then used to compare this vector to host organism *eco_j*. If the host organism is fitter than *eco_{parasite}*, the host organism will survive to the next iteration and *eco_{parasite}* will be eliminated. Conversely, a fitter *eco_{parasite}* will lead to its retention into the next iteration and elimination of *eco_j*.

Updating the Best Organism

When the fitness of the organism *eco_i* is better than the fitness of the *eco_{best}*, the *eco_{best}* is updated with *eco_i*.

Termination

If the current *eco_i* is not the last member of the ecosystem, the SOS will automatically select the next organism to simulate the mutualism, commensalism, and parasitism, and update the eco_{best}. After all members of the ecosystem finish the whole process, SOS will check the termination criteria. The common termination criteria used in the literature are the maximum number of iterations and the maximum number of function evaluations. SOS will stop if one of the termination criteria is reached; otherwise, SOS will start the new iteration.

Practical Examples on Civil Engineering Problems

This section uses three widely used civil engineering problems to assess SOS performance. Obtained SOS optimization results are then compared to data published in the literature. These problems are: (1) reinforced concrete beam design minimization, (2) 25-bar transmission tower truss weight minimization, and (3) site layout optimization for caisson structure fabrication.

Reinforced Concrete Beam Design Minimization

This case study is a cost minimization problem of the reinforced concrete beam as illustrated in Figure 1. This was first presented by Amir and Hasegawa [24]. The beam is assumed simply supported with a 9.144-m (30-ft) span and subject to a live load of 1 ton (2.0 klbf) and a dead load of 0.5 ton (1.0 klbf) accounting for the beam weight. Concrete compressive strength (σ_c) and reinforcing steel yield stress (σ_y) is 34.474 MPa (5 ksi) and 344.74 MPa (50 ksi), respectively. The unit cost of steel and concrete are \$472.4/m^2/ linear m (\$1.0/in^2/linear ft) and \$9.449/-m^2/linear m (\$0.02/in^2/linear ft), respectively. The cross sectional area of reinforcing (A_s), beam width (b), and beam depth (h) are selected as the decision variables.

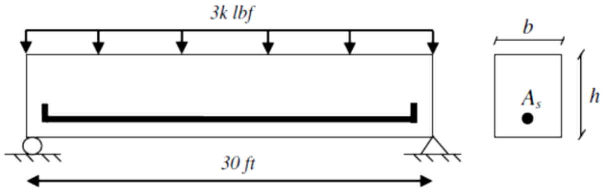

Figure 1. Reinforced Concrete Beam Problem

A_s is determined as a discrete variable and must be chosen from the following list: A_s = [6.0, 6.16, 6.32, 6.6, 7.0, 7.11, 7.2, 7.8, 7.9, 8.0, 8.4] in^2; b is determined as an integer variable: b = [28, 29, 30, … , 39, 40] in; and h is a continuous variable with the boundary limit: $5 \leq h \leq 10$ in.

The structure should be designed to meet the minimal strength required under ACI 318-77 building code:

$$M_u = 0.9A_s\sigma_y(0.8)(1.0 - 0.59\frac{A_s\sigma_y}{0.8bh\sigma_c}) \geq 1.4M_d + 1.7M_l \quad (7)$$

where M_u, M_d, and M_l, respectively, are the flexural strength, dead load, and live load moments of the beam. In this case, M_d = 152.53 kNm (1350 in kip) and M_l = 305.06 kNm (2700 in kip). Beam depth ratio is restricted to be less than or equal to 4. The optimization problem may be stated as:

Minimize: $f(A_s, b, h) = 29.4A_s + 0.6bh$ (8)

Subject to:

$$g_1 = \frac{b}{h} \leq 4 \quad (9)$$

$$g_2 = 180 + 7.375\frac{A_s^2}{h} \leq A_s b \quad (10)$$

Table 1 presents the optimum designs of this problem and the parameters used, including several comparisons with prior research on SD-RC [24], GA and FLC-AHGA [25], CS [26], FA [27]. In this case study, SOS found the same optimum solution identified by FA in 1/10th the time required by FA using 15 organisms.

A 25-bar Transmission Tower Truss Weight Minimization

Over last decades, the 25-bar transmission tower spatial truss (shown in Figure 2) is one of the most studied problems in the field of structural engineering optimization. The structure is composed of 25 members and categorized into 8 groups which are: (1) **A**$_1$, (2) **A**$_2$–**A**$_5$, (3) **A**$_6$–**A**$_9$, (4) **A**$_{10}$–**A**$_{11}$, (5) **A**$_{12}$–**A**$_{13}$, (6) **A**$_{14}$–**A**$_{17}$, (7) **A**$_{18}$–**A**$_{21}$ and (8) **A**$_{22}$–**A**$_{25}$. The members were constructed from materials with a

Table 1. Results of the Reinforced Concrete Beam Example

	Amir and Hasegawa [24] SD-RC	Yun [25]		Gandomi et al. [26] CS	Gandomi et al. [27] FA	Present study SOS
		GA	GA-FL			
A_s (in^2)	7.8	7.2	6.16	6.32	6.32	6.32
b (in)	31	32	35	34	34	34
h (in)	7.79	8.0451	8.75	8.5	8.5	8.5
g_1	-0.0205	-0.0224	0	0	0	0
g_2	-4.2012	-2.8779	-3.6173	-0.2241	-0.2241	-0.2241
f_{min} (in^2)	374.2	366.1459	364.8541	359.2080	359.2080	359.2080
Average	N.A.	N.A.	N.A.	N.A.	460.706	359.7726
Standard deviation	N.A.	N.A.	N.A.	N.A.	80.7387	1.2832
No. of evaluations	396	N.A.	100,000	N.A.	25,000	2,500

Note: 1 in^2 = 6.425 cm^2.

mass density of 2767.99 kg/m^3 (0.1 lb/in.^{3}) and an elastic modulus (E) of 68.95 MPa (10,000 ksi). All members were subjected to stress limitations of ±275.8 MPa (40,000 psi) while all nodes were subject to displacement limitations of ± 0.0226 cm (0.35 in).

Loads are shown in Table 2. There are two types of given variables for this problem. The first version uses discrete variables, while the second version uses continuous variables.

Table 2. Load Case for the 25-bar Spatial Truss Structure

Nodes	Loads		
	P_x (kips)	P_y (kips)	P_z (kips)
1	1.0	−10.0	−10.0
2	0.0	−10.0	−10.0
3	0.5	0.0	0.0
6	0.6	0.0	0.0

Note: 1 kips = 4.448 kN

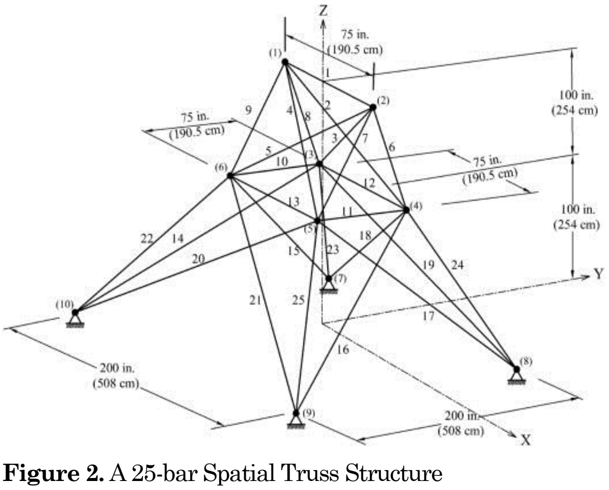

Figure 2. A 25-bar Spatial Truss Structure

For the discrete-problem version, discrete variables are selected from the set:
D = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1. 2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2 .6, 2.8, 3.0, 3.2, 3.4] (in.^{2}). Table 3 summarizes the results obtained by the SOS algorithm and the other optimization methods reported in the literature. HPSO obtained a minimum weight of 219.92 kg (484.85 lbs) in over 25,000 structural analyses. SOS algorithm obtained the same in 20,000 structural analyses with population size of 50.

Table 3. Optimum Design Comparison for the Discrete 25-Bar Spatial Truss Structure

Variables (in^2)	GA [28]	HS [29]	HPSO [30]	SOS
A_1	0.1	0.1	0.1	0.1
A_2–A_5	1.8	0.3	0.3	0.3
A_6–A_9	2.3	3.4	3.4	3.4
A_{10}–A_{11}	0.2	0.1	0.1	0.1
A_{12}–A_{13}	0.1	2.1	2.1	2.1
A_{14}–A_{17}	0.8	1.0	1.0	1.0
A_{18}–A_{21}	1.8	0.5	0.5	0.5
A_{22}–A_{25}	3.0	3.4	3.4	3.4
Weight (lbs)	546.01	484.85	484.85	484.85

Note: 1 in^2 = 6.425 cm^2, 1 lb = 4.448 N.

Table 4 outlines the performance of the SOS algorithm and the other optimization methods reported in the literature for the continuous-problem version. SOS used 50 organisms and 20,000 structural analyses.

The result for the SOS was found after 30 independent runs. The results for the other algorithms were referenced from Degertekin and Hayalioglu [35]. It is apparent that the design solution obtained by HS [31] is theoretically infeasible because these solutions violate the design constraint stated in [35]. The results produced by the SOS algorithm were competitive with those produced by TLBO [35] and SAHS [34] and superior to those of HPSO [32], and BB-BC [33]. Furthermore, the SOS algorithm delivered a better average solution, and lower standard deviation compared to the TLBO algorithm, supporting that the SOS algorithm is a better optimization method than TLBO in terms of consistency.

Site Layout Optimization for Caisson Structure Fabrication

The performance of SOS was validated for solving construction site-level facility layout, a function within the field of construction management. Next, a real-life site-level layout problem previously posited by Kim et al. [36] was investigated. The aim of this case study was to design the site layout for caisson structure fabrication. The site layout considered nine

Table 4. Optimum Design Comparison for the Continuous 25-bar Spatial Truss Structure

Variables (in^2)	HS [31]	HPSO [32]	BB-BC [33]	SAHS [34]	TLBO [35]	SOS
A_1	0.047	0.010	0.010	0.010	0.0100	0.0100
A_2–A_5	2.022	1.970	2.092	2.074	2.0712	1.9848
A_6–A_9	2.950	3.016	2.964	2.961	2.9570	2.9954
A_{10}–A_{11}	0.010	0.010	0.010	0.010	0.0100	0.0100
A_{12}–A_{13}	0.014	0.010	0.010	0.010	0.0100	0.0100
A_{14}–A_{17}	0.688	0.694	0.689	0.691	0.6891	0.6810
A_{18}–A_{21}	1.657	1.681	1.601	1.617	1.6209	1.6784
A_{22}–A_{25}	2.663	2.643	2.686	2.674	2.6768	2.6651
Weight (lbs)	544.38	545.19	545.38	545.12	545.09	545.180
Constraint tolerance (%)	0.206	None	None	None	None	None
Average Weight (lbs)	N/A	N/A	545.78	545.94	545.51	545.292
Standard deviation (lbs)	N/A	N/A	0.491	0.91	0.42	0.102
No. of structure analyses	15,000	125,000	20,566	9,051	15,318	20,000

Note: 1 in^2 = 6.425 cm^2, 1 lb = 4.448 N.

facilities including: (1) steel plate storage, (2) concrete mold storage, (3) steel rod storage, (4) concrete curing place, (5) fabrication factory of caisson wall, (6) prefabrication factory of base plate, (7) steel rod factory, (8) crane 1, and (9) crane 2.

These nine predetermined facilities must be properly assigned to nine predetermined locations scattered over the site. The goal of this case study is to obtain the optimum layout which has the shortest total traveling distance between facilities. The total traveling distance (TD) minimization problem is stated as:

$$\text{Minimize: } \sum_{i=1}^{n} \sum_{x=1}^{n} \sum_{j=1}^{n} \delta_{xi} \text{ x } f_{xi} \text{ x } d_{ij} \qquad (11)$$

Subject to:

$$\sum_{x=1}^{n} \delta_{xi} = 1 \text{, } i = 1, 2, 3, \ldots, n \qquad (12)$$

where n is the number of facility locations; δ_{xi} is the permutation matrix variable such that when facility x is assigned to location I, f_{xi} is the traveling frequency of the construction crew between facilities x and I and d_{ij} is the distance between location i and j. The traveling frequency and distance table are shown in Table 5 and Table 6, respectively.

In this experiment, we compared SOS with PSO and DE. Because the site-level facility layout is a permutation problem, we modified the continuous-based initial solution vector into the permutation vector using the indices that would sort the corresponding initial solution vector. The experiment setup was as follows: All the algorithms used the same common control parameters with a population size of 50 and a total of 20,000 function evaluations. The crossover rate (CR) and the scaling factor (F) for DE were chosen as 0.9 and 0.5, respectively. The cognitive and social factors (c_1 and c_2) were set to 1.8 and the inertia weight (w) was set to 0.6 for PSO. Table 7 summarizes the results obtained by the SOS algorithm and by the other algorithms over 100 independent runs. The best-known answer for this case study is [9 1 8 7 6 5 3 2 4] with a total travel distance of 7727 meters. SOS algorithm delivered the best average solution, worst solution, and lower standard deviation in comparison with DE and PSO. Furthermore, SOS achieved the highest success rate in finding the best solution over 100 runs.

Table 5. Traveling Frequencies between Two Locations

Location	1	2	3	4	5	6	7	8	9
1	0	5	2	2	1	1	4	1	2
2	5	0	2	5	1	2	7	8	2
3	2	2	0	7	4	12	9	4	5
4	2	5	7	0	20	7	8	1	8
5	1	1	4	20	0	30	4	10	3
6	1	2	12	7	30	0	5	8	15
7	4	7	9	8	4	5	0	7	6
8	1	8	4	1	1	8	7	0	9
9	2	2	5	8	3	15	6	9	0

Table 6. Distance between Two Locations (m)

Location	1	2	3	4	5	6	7	8	9
1	0	15	25	33	40	42	47	55	35
2	15	0	10	18	25	27	32	42	50
3	25	10	0	8	15	17	22	32	52
4	33	18	8	0	7	9	14	24	44
5	40	25	15	7	0	2	7	17	37
6	42	27	17	9	2	0	5	15	35
7	47	32	22	14	7	5	0	10	30
8	55	42	32	24	17	15	10	0	20
9	35	50	52	44	37	35	30	20	0

Table 7. Result of Site-level Facility Layout for Caisson Structure

	DE	PSO	SOS
Best (m)	7727	7727	7727
Mean (m)	7769.53	7916.55	7734.90
Worst (m)	8304	8579	7863
Standard deviation (m)	99.42	215.77	23.80
Success Rate	74/100	32/100	89/100

Conclusion

This paper introduced the use of a new optimization algorithm called Symbiotic Organisms Search (SOS) in civil engineering applications. SOS is a population based nature-inspired algorithm that mimics the interactive behavior between organisms in an ecosystem. The three phases of mutualism, commensalism, and parasitism inspire SOS to find the optimal solution for a given objective. Incorporating the characteristic of natural organism interactions into the search strategy supported the superior performance of the SOS algorithm.

In this paper, we first validate the performance of SOS against different optimization methods in constrained benchmark problems and then test the performance of SOS in numerous practical civil engineering problems. SOS precisely identified all optimum solutions in every run with significantly fewer function evaluations than algorithms tested in previous works. The novel SOS algorithm presented in this paper is adequately robust to solve various civil engineering problems. The proposed model may be an effective new tool to guide and support the decision-making process of practitioners.

References

1. Nahry, Tjahjono, T., and Satiti, Y.J., The Optimization Model of Runway and Gate Assignment, *Civil Engineering Dimension*, 15(2), 2013, pp. 129-136.
2. Liao, T.W., Egbelu, P.J., Sarker, B.R., and Leu, S.S., Metaheuristics for Project and Construction Management – A State-of-the-art Review, *Automation in Construction*, 20(5), 2011, pp. 491-505.
3. Osman, I.H. and Laporte, G., Metaheuristics: A Bibliography, *Annals of Operations Research*, 63, 1996, pp. 513-623.
4. Holland, J.H., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.
5. Kennedy, J. and Eberhart, R., Particle Swarm Optimization, *Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia, 1995, pp. 1942-1948.
6. Storn, R. and Price, K., Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization*, 11(4), 1997, pp. 341-359.
7. Karaboga, D. and Basturk, B., A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm, *Journal of Global Optimization*, 39(3), 2007, pp. 459-471.
8. Xu, J. and Li, Z., Multi-Objective Dynamic Construction Site Layout Planning in Fuzzy Random Environment, *Automation in Construction*, 27(0), 2012, pp. 155-169.
9. Yang, I.-T., Using Elitist Particle Swarm Optimization to Facilitate Bicriterion Time-Cost Trade-Off Analysis, *Journal of Construction Engineering and Management*, 133(7), 2007, pp. 498-505.
10. Ponz-Tienda, J.L., Yepes, V., Pellicer, E., and Moreno-Flores, J., The Resource Leveling Problem with Multiple Resources using an Adaptive Genetic Algorithm, *Automation in Construction*, 29, 2013, pp. 161-172.
11. Cheng, M.-Y., Prayogo, D., Wu, Y.-W., and Lukito, M.M., A Hybrid Harmony Search Algorithm for Discrete Sizing Optimization of Truss Structure, *Automation in Construction*, 69, 2016, pp. 21-33.
12. Cheng, M.-Y. and Prayogo, D., A Novel Fuzzy Adaptive Teaching–learning-based Optimization (FATLBO) for Solving Structural Optimization Problems, *Engineering with Computers*, 33(1), 2017, pp. 55-69.
13. Camp, C.V. and Huq, F., CO_2 and Cost Optimization of Reinforced Concrete Frames using a Big Bang-big Crunch Algorithm, *Engineering Structures*, 48, 2013, pp. 363-372.
14. Cheng, Y.M., Li, L., and Fang, S.S., Improved Harmony Search Methods to Replace Variational Principle in Geotechnical Problems, *Journal of Mechanics*, 27(01), 2011, pp. 107-119.
15. Cheng, M.Y. and Prayogo, D., Modeling the Permanent Deformation Behavior of Asphalt Mixtures using a Novel Hybrid Computational Intelligence, *ISARC 2016 - 33rd International Symposium on Automation and Robotics in Construction*, 2016.
16. Cheng, M.-Y., Prayogo, D., and Wu, Y.-W., Novel Genetic Algorithm-based Evolutionary Support Vector Machine for Optimizing High-Performance Concrete Mixture, *Journal of Computing in Civil Engineering*, 2013, pp. 06014003.
17. Cheng, M.-Y., Firdausi, P.M., and Prayogo, D., High-performance Concrete Compressive Strength Prediction using Genetic Weighted Pyramid Operation Tree (GWPOT), *Engineering Applications of Artificial Intelligence*, 29, 2014, pp. 104-113.
18. Cheng, M.-Y. and Prayogo, D., Symbiotic Organisms Search: A New Metaheuristic Optimization Algorithm, *Computers & Structures*, 139, 2014, pp. 98-112.
19. Tejani, G.G., Savsani, V.J., and Patel, V.K., Adaptive Symbiotic Organisms Search (SOS) Algorithm for Structural Design Optimization, *Journal of Computational Design and Engineering*, 2016.
20. Tran, D.-H., Cheng, M.-Y., and Prayogo, D., A Novel Multiple Objective Symbiotic Organisms Search (MOSOS) for Time–cost–labor Utilization Tradeoff Problem, *Knowledge-Based Systems*, 94, 2016, pp. 132-145.

21. Cheng, M.-Y., Chiu, C.-K., Chiu, Y.-F., Wu, Y.-W., Syu, Z.-L., Prayogo, D., and Lin, C.-H., SOS Optimization Model for Bridge Life Cycle Risk Evaluation and Maintenance Strategies, *Journal of the Chinese Institute of Civil and Hydraulic Engineering*, 26(4), 2014, pp. 293-308.
22. Panda, A. and Pani, S., A Symbiotic Organisms Search Algorithm with Adaptive Penalty Function to Solve Multi-objective Constrained Optimization Problems, *Applied Soft Computing*, 46, 2016, pp. 344-360.
23. Deb, K., An Efficient Constraint Handling Method for Genetic Algorithms, *Computer Methods in Applied Mechanics and Engineering*, 186(2–4), 2000, pp. 311-338.
24. Amir, H. and Hasegawa, T., Nonlinear Mixed-Discrete Structural Optimization, *Journal of Structural Engineering*, 115(3), 1989, pp. 626-646.
25. Yun, Y.S., *Study on Adaptive Hybrid Genetic Algorithm and Its Applications to Engineering Design Problems*, 2005, Waseda University.
26. Gandomi, A.H., Yang, X.-S., and Alavi, A.H., Cuckoo Search Algorithm: A Metaheuristic Approach to Solve Structural Optimization Problems, *Engineering with Computers*, 2011, pp. 1-19.
27. Gandomi, A.H., Yang, X.-S., and Alavi, A.H., Mixed Variable Structural Optimization using Firefly Algorithm, *Computers & Structures*, 89(23–24), 2011, pp. 2325-2336.
28. Ringertz, U.L.F.T., On the Methods for Discrete Structural Optimization, *Engineering Optimization*, 13(1), 1988, pp. 47-64.
29. Lee, K.S., Geem, Z.W., Lee, S.-H., and Bae, K.-W., The Harmony Search Heuristic Algorithm for Discrete Structural Optimization, *Engineering Optimization*, 37(7), 2005, pp. 663-684.
30. Li, L.J., Huang, Z.B., and Liu, F., A Heuristic Particle Swarm Optimization Method for Truss Structures with Discrete Variables, *Computers & Structures*, 87(7–8), 2009, pp. 435-443.
31. Lee, K.S. and Geem, Z.W., A New Structural Optimization Method Based on the Harmony Search Algorithm, *Computers & Structures*, 82(9–10), 2004, pp. 781-798.
32. Li, L.J., Huang, Z.B., Liu, F., and Wu, Q.H., A Heuristic Particle Swarm Optimizer for Optimization of Pin Connected Structures, *Computers & Structures*, 85(7–8), 2007, pp. 340-349.
33. Camp, C.V., Design of Space Trusses using Big Bang-big Crunch Optimization, *Journal of Structural Engineering*, 133(7), 2007, pp. 999-1008.
34. Degertekin, S.O., Improved Harmony Search Algorithms for Sizing Optimization of Truss Structures, *Computers & Structures*, 92-93, 2012, pp. 229-241.
35. Degertekin, S.O. and Hayalioglu, M.S., Sizing Truss Structures using Teaching-learning-based Optimization, *Computers & Structures*, 2013.
36. Kim, C.Y., Park, T.I., Ahn, S.M., and Kim, H., Site Layout Optimization for Caisson Structure Fabrication, in *Proceedings of the 29th International Symposium on Automation and Robotics in Construction*, 2012, Eindhoven, The Netherlands.