

The binary salp swarm algorithm with using transfer functions

E.KAYA¹, A.C.CINAR², O.UYMAZ³, S.KORKMAZ⁴, M.S.KIRAN⁵

¹ Dept. of Comp. Eng., Fac. of Eng. Selcuk University, Konya/Turkey, ersinkaya@selcuk.edu.tr

² Turkish State Meteorological Service Konya Division, Konya/Turkey, ahmetcevahircinar@gmail.com

³ Dept. of Comp. Eng., Fac. of Eng. Selcuk University, Konya/Turkey, ouymaz@selcuk.edu.tr

⁴ Dept. of Comp. Eng., Fac. of Eng. Selcuk University, Konya/Turkey, sedatkorkmaz@selcuk.edu.tr

⁵ Dept. of Comp. Eng., Fac. of Eng. Selcuk University, Konya/Turkey, mskiran@selcuk.edu.tr

Abstract - The Salp Swarm Algorithm (SSA) is one of the recently proposed nature-inspired metaheuristic algorithms. SSA mimics the life cycle of salp swarms. Salp swarm is an animal group which lived in oceans. The navigating and foraging behaviors are the characteristic properties of the salp swarms. These behaviors are modeled as an optimization algorithm in SSA and it is firstly proposed for solving continuous optimization problems. In literature, there is no binary version of this algorithm which uses transfer functions. In this work, SSA is modified for solving binary optimization problems by using transfer functions. Transfer functions are used to convert the continuous decision variables to the binary decision variables. With this modification, the structure of SSA has not been changed, but only the Sigmoid and the Tangent Hyperbolic transfer functions are adapted. In order to validate the performance of the proposed binary SSA, a well-known pure binary optimization problem, uncapacitated facility location problems (UFLP), set is considered. UFLPs are used for a benchmarking of many metaheuristic algorithms such as; artificial bee colony, tree-seed algorithm, particle swarm optimization, differential evolution and artificial algae algorithm. The experimental results of 12 UFLPs are compared with each other and state-of-art algorithms. Experimental results demonstrate that the SSA is a promising solver for lower dimensional problems, but its performance should be improved on higher dimensional problems.

Keywords – binary optimization, salp swarm algorithm, swarm intelligence, transfer functions, logistic functions.

I. INTRODUCTION

Metaheuristic optimization algorithms are trending topic in the numeric optimization research field. Almost every month a new nature-inspired metaheuristic algorithm is proposed. The Salp Swarm Algorithm (SSA) is a population-based metaheuristic algorithm which mimics the navigating and foraging behaviors of salps [1]. This algorithm is in the swarm intelligence algorithms group. Mirjalili et al. [1] proposed the SSA for solving the single objective continuous optimization problems. Optimization problems are divided into two main groups according to the type of decision variables. The names of these groups are continuous and discrete. Binary optimization is a subfield of discrete optimization. In literature, there are many binary optimization algorithms [2-8] in order to effectively solve this type of problems. According to the literature review, there is no binary version of the SSA. In this work, we propose

a new binary SSA for solving binary optimization problems by using transfer functions. Transfer functions convert the continuous decision variables to binary form. Thus, any modification is not required the structure of the algorithm, but only decision variables are transformed by transfer functions to binary solution space. In this work, we use sigmoid and tangent hyperbolic transfer functions in order to achieve this purpose. The log-sigmoid and tangent hyperbolic transfer functions produce values in range of [0, 1] and [-1,1] by depending on the input, respectively. The input range for these functions [-6, 6] is considered and the values in this range are converted to the binary variables by using the transfer functions. The uncapacitated facility location problem (UFLP) is a pure binary optimization problem and there is no integer or real number in this problem. A lot of combinatorial optimization problems such as set covering, set partitioning, airline crew scheduling [3-8] may be modelled by way of UFLP and thus, the solving the UFLP is an important attempt in binary optimization field. The rest of the paper is organized as follows. Section 2 gives information about SSA. Binary SSA is explained in the Section 3. Experimental study is conducted in Section 4 and Section 5, respectively. Finally, the obtained results are discussed in Section 6.

II. SALP SWARM ALGORITHM (SSA)

A salp is an animal which lives in water and similar to jellyfishes. The swarming behavior of salps is modeled by Mirjalili et al. in 2017 [1] for an optimization algorithm. Salp Swarm Algorithm (SSA) is a population-based metaheuristic algorithm which mimics the navigating and foraging behaviors of salps. The salp swarm is named as salp chain in this algorithm. The single salp is shown in Figure 1(a) and the swarm of salps is shown in Figure 1(b).

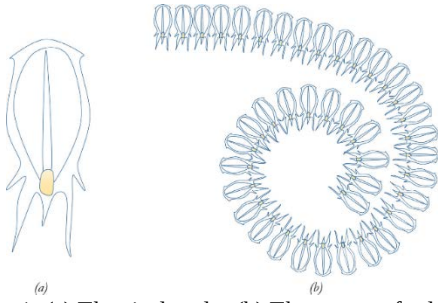


Figure 1: (a) The single salp, (b) The swarm of salps [1]

The salp population is divided into two main groups: leader and followers. The salps search the food source in the search space by using these behaviors. The position of the leader salp updates as follows:

$$x_j^1 = \begin{cases} F_j + c_1 ((UB_j - LB_j) \times c_2) + LB_j & c_3 \geq 0 \\ F_j - c_1 ((UB_j - LB_j) \times c_2) + LB_j & c_3 < 0 \end{cases} \quad (1)$$

where x_j^1 shows the position of leader salp in the j th dimension, F_j is the position of the food source in the j th dimension, UB_j states the upper bound of j th dimension, LB_j states the lower bound of j th dimension, c_1 , c_2 , and c_3 are random numbers. The parameters c_2 and c_3 are uniform random numbers which are generated between 0 and 1.

The parameter c_1 balances exploration and exploitation as follows:

$$c_1 = 2e^{-\left(\frac{4 \times iter}{Maxiter}\right)^2} \quad (2)$$

where $iter$ is the current iteration number and $Maxiter$ is the maximum number of iterations. The real salps use the following the model for movement.

$$x_j^i = \frac{1}{2}at^2 + v_0t \quad (3)$$

where $i \geq 2$, x_j^i shows the position of i th follower salp in j th dimension, t is time, v_0 is the initial speed, and $a = \frac{v_{final}}{v_0}$ where $v = \frac{x - x_0}{t}$. For simplicity, this formula was converted to Eq.4.

$$x_j^i = \frac{1}{2}(x_j^i - x_j^{i-1}) \quad (4)$$

The pseudo code of the SSA is given in Figure 2.

```

The salp population are randomly created between upper and lower bounds
 $x_i$  ( $i = 1, 2, \dots, n$ )
while (termination criterion is not met)
    Calculate the objective function value of each salp
    LeaderSalp = the best salp
    Update  $c_1$  by Eq.2
    for each salp ( $x_i$ )
        if ( $i == 1$ )
            Update the position of the leader salp by Eq.1
        else
            Update the position of the follower salps by Eq.4
        end
    end
    Salps outside the search space are restricted.
end
return LeaderSalp
    
```

Figure 2: Pseudo code of the Salp Swarm Algorithm

III. BINARY SALP SWARM ALGORITHM (BSSA)

The sigmoid and tangent hyperbolic transfer functions are used for binarization of the SSA. Continuous decision variables converted to the binary form with the transfer functions. In this way, no structural change is made to the algorithm. Before the calculating objective function values, continuous variables converted to binary variables. The continuous variables are created between -6 and 6, which are the upper and lower bound of the search space of the problem. Thus, the output of sigmoid function is between 0.0025 and 0.9975. Eq.5 shows how sigmoid transfer function (S) works in BSSA.

$$transfer(x_{i,j}) = \begin{cases} binX_{i,j} = 1, & \text{if } rand(0,1) < S(X_{i,j}) \\ binX_{i,j} = 0, & \text{otherwise} \end{cases} \quad (5)$$

The tangent hyperbolic transfer function is create values between -1 and 1. Eq.6 shows how tangent hyperbolic transfer function (TH) is used for creating 0 or 1 in BSSA.

$$transfer(x_{i,j}) = \begin{cases} binX_{i,j} = 1, & \text{if } rand(0,1) < TH(X_{i,j}) \\ binX_{i,j} = 0, & \text{otherwise} \end{cases} \quad (5)$$

The mathematical formula of sigmoid logistic function is shown in Eq.7 and the mathematical formula of the tangent hyperbolic logistic function is shown in Eq. 8.

$$S(x) = \frac{1}{1 + e^{-x}} \quad (7)$$

$$TH(x) = \frac{2}{1 + e^{-2x}} - 1 \quad (8)$$

IV. UNCAPACITATED FACILITY LOCATION PROBLEM

The uncapacitated facility location problem (UFLP) is a pure binary optimization problem. In this problem, there are facilities, the customers, the logistic operations cost and the setting up cost. The main objective is to minimize the sum of these costs. For achieving this aim, a minimum number of facilities should be opened in the direction of customer demands.

$F = \{f_1, f_2, \dots, f_m\}$ is the set of the facilities and m is the number of facilities, $C = \{c_1, c_2, \dots, c_n\}$ is the set of customers and n is the number of customers, $S = \{s_f\}$ is the setting up cost of the facility f , $T = [t_{fc}]$ is the transportation costs from facility f to customer c . The objective function is as follows in Eq.9:

$$argmin \left\{ O(X) = \sum_{f \in X} s_f + \sum_{c \in C} \min\{t_{fc} | f \in X\} : \emptyset \subset X \subseteq F \right\} \quad (9)$$

where X is a solution, $O()$ is the objective function. The several combinatorial optimization problems can be modelled by way of the UFLP form. For instance, set covering, set partitioning, airline crew scheduling [3-8]. Therefore, solving the UFLP means solving these problems immediately.

V. BENCHMARK SET

For validating the performance of the BSSA, we use 12 UFLPs which are directly taken from OR Library [9]. The problem names, sizes and optimum values are given in Table 1. The small size problems (cap71, cap72, cap73 and cap74) have

16 facilities and 50 customers. The medium size problems (cap101, cap102, cap103 and cap 104) have 25 facilities and 50 customers. The large size problems (cap131, cap132, cap133 and cap 134) have 50 facilities and 50 customers. The cost of optimal solutions are given in Table 1.

Table 1: The benchmark set

Problem name	Problem size	Cost of optimal solution
Cap71	16x50	932,615.75
Cap72	16x50	977,799.40
Cap73	16x50	1,010,641.45
Cap74	16x50	1,034,976.98
Cap101	25x50	796,648.44
Cap102	25x50	854,704.20
Cap103	25x50	893,782.11
Cap104	25x50	928,941.75
Cap131	50x50	793,439.56
Cap132	50x50	851,495.33
Cap133	50x50	893,076.71
Cap134	50x50	928,941.75

VI. RESULTS AND DISCUSSIONS

The relative error (RE) is used for the determining the performance of transfer functions and population sizes. The mathematical formula of the RE is shown in Eq.10.

$$RE(\%) = \frac{Best - Optimum}{Optimum} \times 100 \quad (10)$$

where, Best is the obtained solution by the method, Optimum is the cost of optimum solution for the problem. The population sizes are taken as 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100. For making a fair comparison, we take $8E+4$ for a maximum number of function evolutions (Max_FEs) as termination condition. The results of the compared algorithms are directly taken from [3, 6, 10]. Table 2 shows the RE values of Sigmoid BSSA.

Table 2: The RE values of Sigmoid BSSA

Problem/N	10	20	30	40	50	60	70	80	90	100
cap71	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00
cap72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
cap73	0.01	0.00	0.01	0.00	0.00	0.01	0.01	0.00	0.02	0.00
cap74	0.00	0.00	0.02	0.00	0.00	0.01	0.00	0.00	0.01	0.01
cap101	0.10	0.10	0.10	0.12	0.13	0.07	0.15	0.13	0.10	0.16
cap102	0.08	0.07	0.09	0.11	0.08	0.13	0.17	0.19	0.14	0.13
cap103	0.07	0.07	0.09	0.17	0.09	0.09	0.10	0.12	0.09	0.17
cap104	0.13	0.15	0.26	0.26	0.14	0.19	0.28	0.40	0.18	0.29
cap131	1.74	1.81	2.25	2.42	1.82	2.49	2.41	2.26	2.20	2.16
cap132	1.51	2.63	2.83	2.84	2.89	2.86	2.17	2.93	2.40	2.50
cap133	2.99	2.37	4.31	3.71	3.21	3.58	3.65	3.25	3.43	2.84
cap134	3.50	4.18	4.84	6.31	6.10	6.66	5.83	6.51	6.68	5.94
Total RE	10.14	11.38	14.79	15.95	14.46	16.08	14.79	15.78	15.26	14.20

According to the Table 2, the small-sized problems (cap71, cap72, cap73 and cap74) are solved optimally, the medium-sized problems (cap101, cap102, cap103 and cap104) are solved with about one percent less error than optimums and the

large-sized problems (cap131, cap132, cap133 and cap134) are solved about five percent less error than optimums. The best population size is 10 according to the total RE. The Table 3 shows the RE values of BSAA based on Tangent Hyperbolic.

Table 3: The RE values of Tangent Hyperbolic BSSA

Problem/N	10	20	30	40	50	60	70	80	90	100
cap71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
cap72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
cap73	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.01	0.00	0.01
cap74	0.00	0.01	0.01	0.01	0.00	0.00	0.01	0.00	0.02	0.00
cap101	0.09	0.11	0.11	0.10	0.09	0.14	0.14	0.13	0.11	0.13
cap102	0.08	0.11	0.12	0.11	0.12	0.10	0.09	0.10	0.15	0.12
cap103	0.07	0.15	0.12	0.13	0.13	0.13	0.08	0.14	0.09	0.10
cap104	0.18	0.15	0.18	0.26	0.26	0.22	0.17	0.48	0.23	0.14
cap131	1.33	1.63	2.04	1.68	1.47	1.51	1.31	1.50	1.56	1.41
cap132	1.70	1.63	1.89	1.73	1.98	1.81	1.91	2.03	1.83	1.62
cap133	1.55	2.27	2.35	2.10	2.19	2.03	2.42	2.34	2.05	1.85
cap134	2.42	3.34	3.41	3.96	3.97	4.10	3.76	3.63	3.25	3.49
Total RE	7.43	9.38	10.24	10.07	10.21	10.04	9.89	10.36	9.30	8.88

According to the Table 3, the small and medium-sized problems are solved about optimally and the large size problems are solved about three percent less error than optimums. The best population size is 10 according to the total RE in Tangent Hyperbolic BSSA same as the Sigmoid BSSA. Additionally, according to the total RE values, the usage of tangent hyperbolic is better than the usage of sigmoid as a transfer function in BSSA.

The results of BSSA are compared with the results of binary variants of ABC, PSO, TSA and DE algorithms and the comparison results are given in Table 4.

Table 4: The comparison of state-of-art algorithms

Problem/N	BSSA-TanH	BPSO	SimLogicTSA	IBPSO	binABC	disABC	disDE/rand	binDE
cap71	0.00	0.00	0.00	0.04	0.00	0.00	0.00	0.00
cap72	0.00	0.00	0.00	0.27	0.00	0.00	0.00	0.00
cap73	0.00	0.02	0.00	0.20	0.00	0.00	0.00	0.00
cap74	0.00	0.01	0.00	0.40	0.00	0.00	0.00	0.00
cap101	0.09	0.05	0.00	0.60	0.00	0.00	0.00	0.00
cap102	0.08	0.01	0.00	0.73	0.00	0.00	0.00	0.00
cap103	0.07	0.04	0.00	0.64	0.00	0.00	0.01	0.00
cap104	0.18	0.08	0.00	1.00	0.00	0.00	0.00	0.00
cap131	1.33	0.13	0.00	2.42	0.00	0.62	0.00	0.00
cap132	1.70	0.09	0.00	3.60	0.00	0.09	0.00	0.01
cap133	1.55	0.11	0.00	5.26	0.12	0.03	0.01	0.01
cap134	2.42	0.13	0.00	7.63	0.00	0.00	0.00	0.00
Total RE	7.43	0.69	0.00	22.80	0.12	0.75	0.03	0.02

In Table 4, binary salp swarm algorithm with tangent hyperbolic transfer function (BSSA-TanH), binary particle swarm optimization (BPSO) [11], similarity and logic gate-based tree-seed algorithm (SimLogicTSA) [3], improved binary particle swarm optimization (IBPSO) [12], binary artificial bee colony (binABC) [6], similarity-based artificial bee colony (disABC) [4], similarity-based differential evolution with random position update rule (disDE/rand) [10], binary differential evolution (binDE) [10] are compared. The best algorithm is SimLogicTSA and the worst is the IBPSO for solving these UFLPS. The BSSA is a promising solver for

lower dimensional problems, but its performance should be improved on higher dimensional problems.

The convergences of population sizes on cap71, cap101 and cap 131 problems are shown in Figure 3-5, respectively. Due to the page limitation, the convergence graphs of the other problems are not included in this paper.

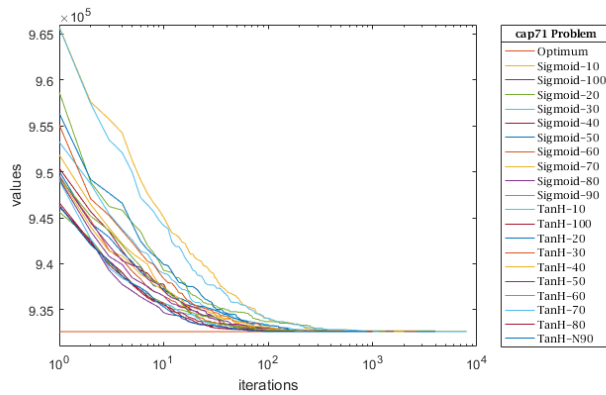


Figure 3: The convergences of population sizes on cap71 problem

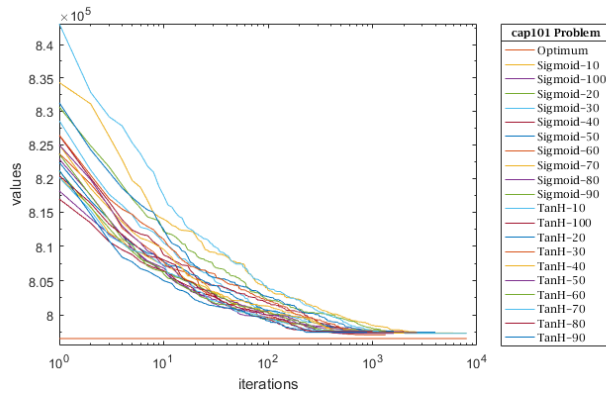


Figure 4: The convergences of population sizes on cap101 problem

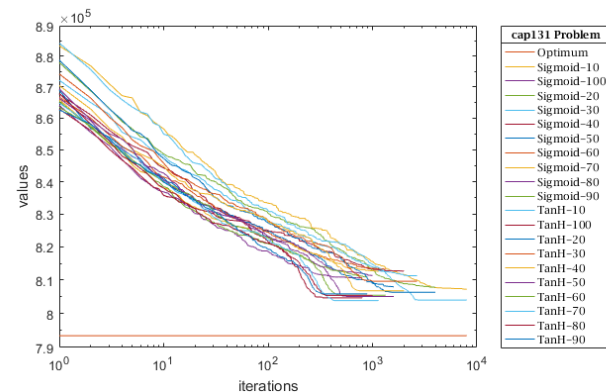


Figure 5: The convergences of population sizes on cap131 problem

The BSSA produces comparable solutions on small and medium-sized UFLPs, but for large instances BSSA produces non-optimal solutions and show lower performance when compared to state-of-art algorithms. To overcome this problem, different binarization mechanisms should be integrated with this algorithm.

REFERENCES

1. Mirjalili, S., et al., *Salp swarm algorithm: a bio-inspired optimizer for engineering design problems*. Advances in Engineering Software, 2017. **114**: p. 163-191.
2. Banitalebi, A., M.I.A. Aziz, and Z.A. Aziz, *A self-adaptive binary differential evolution algorithm for large scale binary optimization problems*. Information Sciences, 2016. **367**: p. 487-511.
3. Cinar, A.C. and M.S. Kiran, *Similarity and logic gate-based tree-seed algorithms for binary optimization*. Computers & Industrial Engineering, 2018. **115**: p. 631-646.
4. Kashan, M.H., N. Nahavandi, and A.H. Kashan, *DisABC: a new artificial bee colony algorithm for binary optimization*. Applied Soft Computing, 2012. **12**(1): p. 342-352.
5. Kiran, M.S., *The continuous artificial bee colony algorithm for binary optimization*. Applied Soft Computing, 2015. **33**: p. 15-23.
6. Kiran, M.S. and M. GÜNDÜZ, *XOR-based artificial bee colony algorithm for binary optimization*. Turkish Journal of Electrical Engineering & Computer Sciences, 2013. **21**(Sup. 2): p. 2307-2328.
7. Korkmaz, S., A. Babalik, and M.S. Kiran, *An artificial algae algorithm for solving binary optimization problems*. International Journal of Machine Learning and Cybernetics, 2017: p. 1-15.
8. Korkmaz, S. and M.S. Kiran, *An artificial algae algorithm with stigmergic behavior for binary optimization*. Applied Soft Computing, 2018.
9. Beasley, J.E., *OR-Library: distributing test problems by electronic mail*. Journal of the operational research society, 1990. **41**(11): p. 1069-1072.
10. Kashan, M.H., A.H. Kashan, and N. Nahavandi, *A novel differential evolution algorithm for binary optimization*. Computational Optimization and Applications, 2013. **55**(2): p. 481-513.
11. Kennedy, J. and R.C. Eberhart, *A discrete binary version of the particle swarm algorithm*. in *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*. 1997. IEEE.
12. Yuan, X., et al., *An improved binary particle swarm optimization for unit commitment problem*. Expert Systems with applications, 2009. **36**(4): p. 8049-8055.