An International Journal of Optimization and Control: Theories & Applications
ISSN: 2146-0957 eISSN: 2146-5703
Vol.10, No.2, pp.188-197 (2020)
https://doi.org/10.11121/ijocta.01.2020.00775

RESEARCH ARTICLE

A modified crow search algorithm for the weapon-target assignment problem

Emrullah Sonuç *

[a] *Department of Computer Engineering, Karabuk University, Turkey*
esonuc@karabuk.edu.tr

ARTICLE INFO

Article history:
Received: 26 January 2019
Accepted: 31 January 2020
Available Online: 4 June 2020

Keywords:
Combinatorial optimization
Crow search algorithm
Nature inspired meta-heuristic algorithms
Weapon-target Assignment Problem

AMS Classification 2010:
68T20, 90C27

ABSTRACT

The Weapon-Target Assignment (WTA) problem is one of the most important optimization problems in military operation research. In the WTA problem, assets of defense aim the best assignment of each weapon to target for decreasing expected damage directed by the offense. In this paper, Modified Crow Search Algorithm (MCSA) is proposed to solve the WTA problem. In MCSA, a trial mechanism is used to improve the quality of solutions using parameter LIMIT. If the solution is not improved after a predetermined number of iterations, then MCSA starts with a new position in the search space. Experimental results on the different sizes of the WTA problem instances show that MCSA outperforms CSA in all problem instances. Also, MCSA achieved better results for 11 out of 12 problem instances compared with four state-of-the-art algorithms. The source codes of MCSA for the WTA are publicly available at http://www.3mrullah.com/MCSA.html

1. Introduction

Weapon-Target Assignment (WTA) problem is one of the most important optimization problems in military operation research. The WTA problem has two versions as the static weapon-target assignment problem (SWTA) and the dynamic weapon-target assignment problem (DWTA). The main difference between the SWTA and the DWTA is the timing of launching weapons to targets. In the DWTA, the launching of weapons is performed asynchronously, however in the SWTA, all weapons are launching at the same time and only once [1]. In the WTA problem, the aim is to minimize the damage caused by attacks of the targets. Hence, assets of the defense aim the best assignments for minimal damage after the engagement. Several exact and approximation algorithms [2–4] have recently involved in solving the WTA problem. Since the WTA is an NP-complete problem [5], exact algorithms can not solve large-scale WTA problems in polynomial time. To overcome this problem, metaheuristic algorithms are presented to solve the WTA problem. Metaheuristic algorithms provide a valid solution in a reasonable time [6].

In recent years, metaheuristic algorithms for solving optimization and engineering problems have attracted much attention in the literature. The development of nature-inspired metaheuristic algorithms has increased rapidly in the last decades [7]. These algorithms have good ability to solve global optimization problems even it is complex or high dimensional. The strategy of metaheuristic algorithms is to obtain a solution in a reasonable time for optimization problems which are naturally intricate and very hard to solve. This strategy is built on two main features: exploration and exploitation. In the exploration stage, the algorithm attempts to find a new solution in the search space. In the exploitation stage, the algorithm searches for the neighborhood of the highest quality solution so far to get better solutions. The balance of these two stages is highly important for the algorithm to be successful. The Crow Search Algorithm (CSA) [8] is a population-based metaheuristic algorithm inspired by the behavior of crows, has a good exploration and exploitation for optimization problems.

Many metaheuristic algorithms have been proposed for the WTA problem. Şahin and Leblebicioğlu [9] presented a Hierarchical Fuzzy Decision Maker method to achieve the best assignment for improving performance on the battlefield. The proposed method increased the approximation performance in comparison to exact and optimal methods. Wang et al. [10] developed a Grey Wolf Optimizer which is the

*Corresponding author

popular population-based algorithm in recent years, to solve the WTA problem. The problem was addressed as a binary problem and the algorithm was modified to a discrete method. According to results, Grey Wolf Optimizer resulted in good quality solutions for small-scale problems and proved that it is competitive for large-scale problems. Li et al. [11] have presented an Ant Colony Optimization for bi-objective the WTA problem. In their study, an optimization model for the WTA is designed which maximizes the expected damage of the enemy (first objective) and minimizes the cost of missiles (second objective). Due to the bi-objective model of the WTA, Ant Colony Optimization is modified to get a set of Pareto solutions. According to simulation results, the modified algorithm improved the performance of the pure one and produced better solutions. Sonuc et al. [12] have worked on a Simulated Annealing algorithm to solve the SWTA problem on GPU. The aim of the study was to obtain better solutions with less computational time compared to the solution of the serial algorithm. Computational results on problem instances have shown that the parallel algorithm was 250 times faster than a single-core CPU and improved the quality of solutions. Zhang et al. [13] have developed a hybrid method using Ant Colony Optimization and Genetic Algorithm to obtain fast convergence speed for the WTA problems. Implementation of Artificial Bee Colony algorithm which is inspired by intelligent behavior of honey bees, was proposed for solving the SWTA problem by Durgut et al. [14]. In the study, three local search operators were discussed and according to the results, the swap operator emerged as more effective than insertion and inversion operators. Kutucu et al. [15] presented a hybrid method with Artificial Bee Colony and Simulated Annealing for the SWTA. According to results on benchmark problems, the proposed algorithm was competitive and satisfactory compared to other metaheuristic algorithms for the WTA. To improve the ability of Ant Colony Optimization, an immune system based algorithm was developed to solve the WTA by Lee et al. [16]. According to the comparison results, the proposed algorithm has improved searching performance. Hu et al. [17] improved Ant Colony Optimization in the viewpoints of selection, updating and concentration interval and applied it to the WTA problem. The advantages of the proposed algorithm were faster convergence and better avoidance from local optima. Tokgöz et al. [18] presented combinatorial optimization techniques for WTA problems. Several heuristic algorithms were selected and applied to the WTA and the results proved that Variable Neighborhood Search and Simulated Annealing obtained better solutions than other algorithms. Li et al. [19] developed a decomposition-based evolutionary algorithm for multiobjective SWTA. According to experiments, the proposed method was effective and promising on generated scenarios. Also, real-time heuristics using Construction Heuristic, Quiz Problem Search Heuristic and Greedy Branch and Bound Heuristic, was presented by Kline et al. [20]. All three heuristics were used for comparison with existing heuristics in literature and the results outlined that the computational costs of the proposed methods are less expensive than the existing ones. Hocaoglu [21] aims to generate a model for air defense. The model answers to the question that is how many missiles are necessary to eliminate attacking from the offense. The model gives a better and faster than the Simulated Annealing algorithm.

This paper aims to improve the quality of solutions for the SWTA problem using a modified crow search algorithm (MCSA). MCSA is a population-based algorithm and obtained better solutions in less time compared to Simulated Annealing [1] which is an iterative heuristic algorithm. Besides, one agent searches a new solution in the search space for each iteration hence Simulated Annealing has a poor exploration compared to population-based metaheuristics. Also, MCSA was compared with the state-of-the-art algorithms and the experimental results were revealed that MCSA was improved quality of results in 6 of 12 problems. The rest of this paper is organized as follows. In Section 2, the model of the SWTA problem is illustrated and the formulation of the problem is presented. In Section 3, nature-inspired CSA is introduced. In Section 4, MCSA based on a trial mechanism is proposed. Experimental results on the WTA problems are presented to demonstrate the performance of improved CSA in Section 5. Finally, conclusion and future works are described in Section 6.

2. Problem formulation

According to the WTA model, which is a minimization optimization problem, assets of defense aim the best assignment of each weapon to target for decreasing expected damage directed by the offense. Each weapon has a destroying probability for each target and the expected damage for assets of defense is evaluated after engagement in the battlefield. An illustration of the WTA problem is presented in Figure 1.

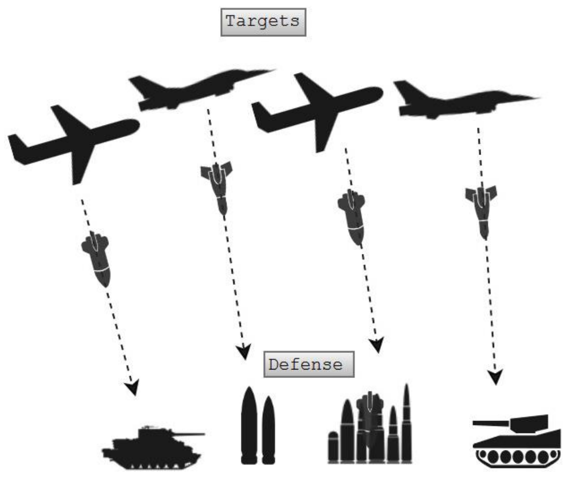


Figure 1. Illustration of the WTA problem.

Table 1 shows the explanation of each symbol for the WTA model. In general, a WTA problem for a defensive mission can be formulated as follows:

$$f(\pi) = \min \sum_{i=1}^{n} v_i \prod_{j=1}^{m} \left(1 - p_{ij}\right)^{x_{ij}} \quad (1)$$

$$s.\,t.\ \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, ..., m. \quad (2)$$

Table 1. Definition of symbols for the WTA model.

Symbol	Explanation
n	the number of targets
m	the number of weapons
v_i	the value of the target i
p_{ij}	the probability of destroying by assigning the weapon j to the target i,
$x = [x_{ij}]$	the decision variable that is nxm matrix, where $x_{ij} = \begin{cases} 1 \text{ if weapon } j \text{ is assigned to target } i, \\ 0 \text{ otherwise} \end{cases}$

3. The crow search algorithm (CSA)

Crows live in flocks and can follow the other birds and steal the food they have stored in their nests. As a result of this follow-up, they can remember the location of other birds' hiding-place and find it whenever they want. The pseudocode of the CSA, which is inspired by the behavior of crows, is shown in Figure 2. CSA has an easy to implement structure and only needs two parameters. Implementation of CSA for optimization problems is an easy process since it has only two parameters: Awareness Probability (*AP*) and Flight Length (*FL*).

According to the strategy of CSA, the crow updates its position in two states. In the first state, each crow (*crow i)* selects a random crow (*crow j*) to steal food from its hiding place without being noticed. The decision to follow the selected crow is determined by the parameter *AP*. If the follow-up is carried out, the new position of the crow is determined according to Eq. (3) using the memory of *crow j* (m_j).

$$x^{i,iter+1} = x^{i,iter} + r_i \,.\, fl^{i,iter} \,.\, (m^{j,iter} - x^{i,iter}) \quad (3)$$

The second state is that *crow j* recognizes that is being followed by *crow i*. In this state, the crow moves to a new position in the search space. For the second state, the new position of the crow is defined as follows:

$$x^{i,iter+1} = \begin{cases} x^{i,iter+1} = x^{i,iter} + r_i \,.\, fl^{i,iter} \,.\, (m^{j,iter} - x^{i,iter}) & r_j \geq AP^{j,iter} \\ a \text{ random position} & otherwise \end{cases} \quad (4)$$

```
Initialize the crows population X_i (i = 1, 2, ..., N)
Evaluate the position of each crow in the search space
Initialize the memory of each crow
while (iter < iter_max)
        for i = 1 : N (all N crows in the population)
                Randomly select one crow to follow (e.g. crow j)
                Set an awareness probability
                if r_j ≥ AP^{j,iter}
                        Update the position of the current crow by the Eq. (3)
                else
                        Generate a new position in the search space for the current crow
                end if
        end for
        Check if any crow goes beyond the search space and amend it
        Evaluate the new position of each crow
        Update the memory of each crow
end while
```

Figure 2. Pseudocode of the CSA.

4. The WTA problem using MCSA

The WTA problem is a combinatorial optimization problem and each weapon must be assigned to a target. This assignment is represented as a permutation in the problem. Also, this permutation represents a position in the search space for a crow. The aim is finding the best position (permutation) in the search space to minimize the objective function (Eq. (1)). CSA is modified to improve the quality of solutions using a new parameter called *LIMIT*. If a solution that represents a position in the search space, is not improved by a predetermined number of trials, then a new position is generated. This method is proposed by Karaboga et al. [22,23] for Artificial Bee Colony Algorithm to solve optimization problems. The implementation of MCSA for the SWTA problem is carried out through the following steps:

Step 1. Initialization of MCSA parameters.

Initialize the parameters: *N*, $iter_{max}$, *FL*, *AP* and number of non-improved trials *LIMIT*.

Step 2. Initialize permutation and memory of crows.

Randomly generate a permutation for each crow and memorize the initial permutations.

Step 3. Evaluate the objective function.

Compute objective function using its permutation for each crow.

Step 4. Generate a new permutation.

Generate a new permutation for *crow i* as follows:

Randomly select one other crow (*crow j*) to use its permutation. Generate a new position using the swap operator (see Figure 3.) for permutation of *crow j*. Thus, a new permutation of crow *i* is determined if $r_j \geq AP^{j,iter}$. This procedure is repeated for all crows. Otherwise, it keeps its current permutation. This procedure is defined as follows:

$$x^{i,iter+1} = \begin{cases} new\ permutation\ with\ swapping & r_j \geq AP^{j,iter} \\ keep\ the\ current\ permutation & otherwise \end{cases} \quad (5)$$

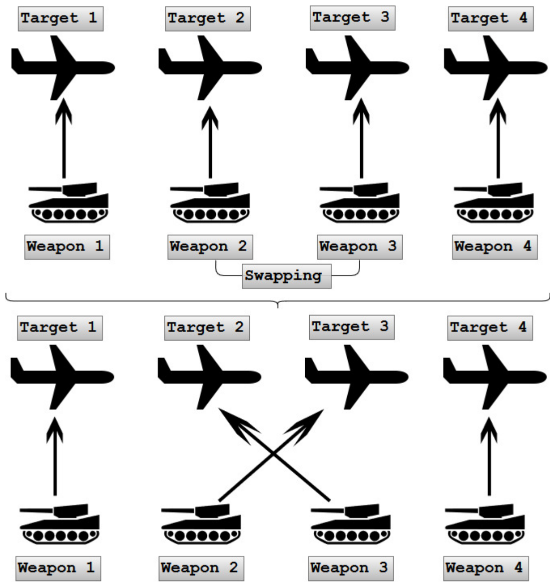


Figure 3. Illustration of swap operator for neighborhood solution.

Step 5. Evaluate the objective function of new permutations.

Compute the objective function of the new permutation for each crow.

Step 6. Update memory.

If the new objective function value of each crow is less than the memorized one, then update the memory of each crow using:

$$m^{i,iter+1} = \begin{cases} x^{i,iter+1} & f(x^{i,iter+1}) < f(m^{i,iter+1}) \\ m^{i,iter+1} & otherwise \end{cases} \quad (6)$$

Step 7. Check if the trial value is reached to LIMIT or not.

After a predetermined number of trials, if there is no improvement on the solutions for the population, generate a new permutation for each crow using the equation is as follows:

$$x^{i,iter+1} = \begin{cases} generate\ a\ random\ permutation & r_i > AP^{i,iter}\ .\ fl^{i,iter} \\ keep\ the\ current\ permutation & otherwise \end{cases} \quad (7)$$

For each crow, the objective function value of the new permutation is computed.

Step 8. Evaluate the objective function and update memory.

Computation of objective function for each crow using its permutation. After computation, update the memory of crows.

Step 9. Check stop criterion.

Repeat Steps 4–8 until $iter_{max}$ is reached.

The flowchart of MCSA is presented in Figure 4.

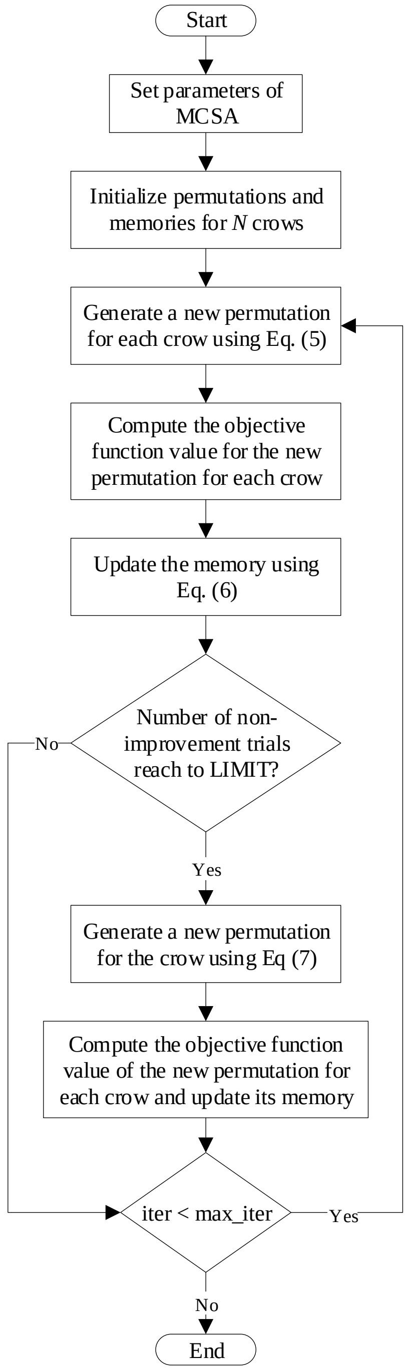


Figure 4. Flowchart of the modified CSA for solving the WTA problem.

5. Experimental results

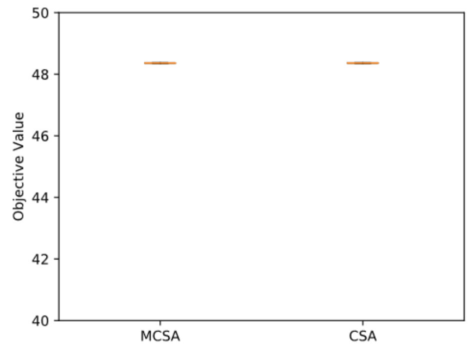
MCSA is tested on 12 problem instances (available at https://doi.org/10.17632/jt2ppwr62p.1) presented in [12]. Dimensions of problem instances are in the range 5 – 200 and listed in Table 2. The numerical experiments were performed on a PC with Intel(R) Core(TM) i7-5600U CPU @ 2.60 GHz, with 8.00 GB of RAM, running Windows 8 64-bit operating system. The codes of MCSA and CSA have been written in C under CodeBlocks IDE v17.12.

5.1. Comparison MCSA and CSA

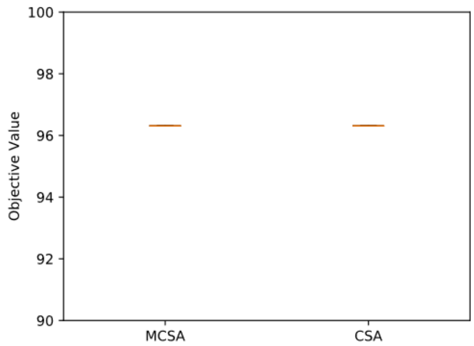
Firstly, robustness of MCSA is tested in comparison with the pure CSA by using parameters which are AP = 0.2, FL = 2, N= 20, $ITERATION$ = 1000 and $LIMIT$ = 10 x size of problem (for MCSA only). Figure 5 shows the box plot of 10 independent runs for the problem instances from WTA1 to WTA12 with the aim of comparison between MCSA and CSA. The results show that MCSA outperforms CSA in all problem instances. Also, the box plots show that MCSA converges quickly to the optimal solutions as it has better values and fewer heights compared to CSA.

Table 2. The WTA problem instances.

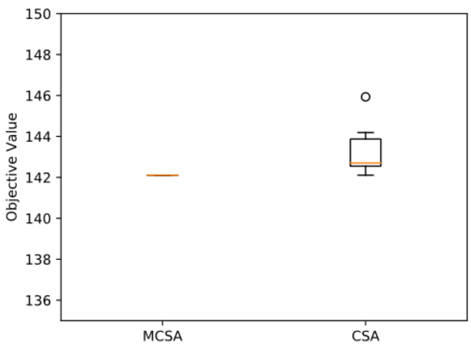
Instance No	Number of Weapons	Number of Targets
#1	5	5
#2	10	10
#3	20	20
#4	30	30
#5	40	40
#6	50	50
#7	60	60
#8	70	70
#9	80	80
#10	90	90
#11	100	100
#12	200	200



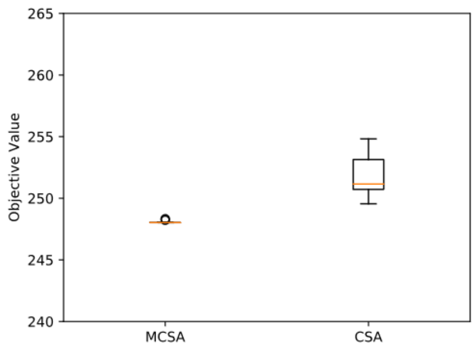
(a) Box plot for WTA1.



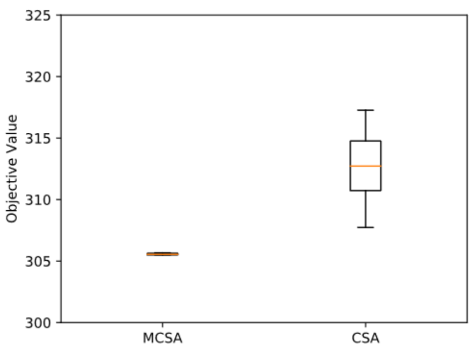
(b) Box plot for WTA2.



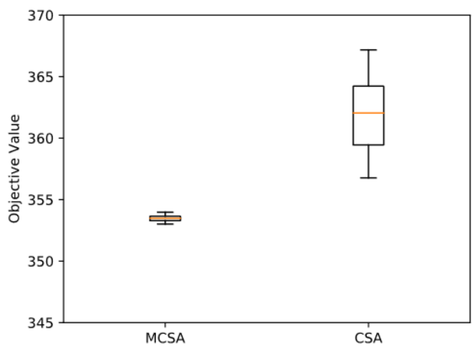
(c) Box plot for WTA3.



(d) Box plot for WTA4.

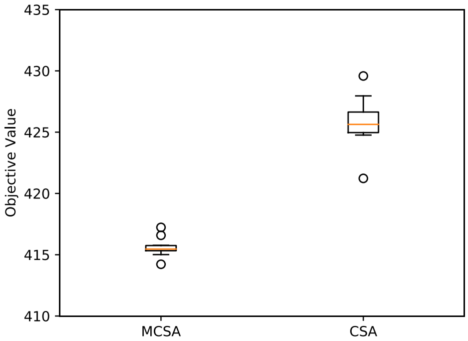


(e) Box plot for WTA5.

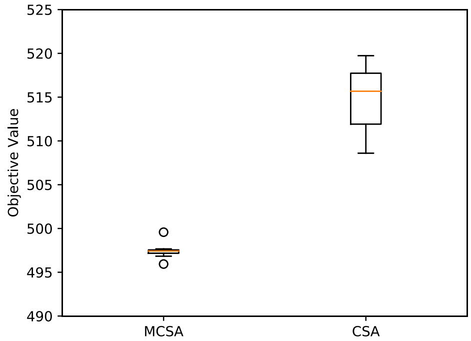


(f) Box plot for WTA6.

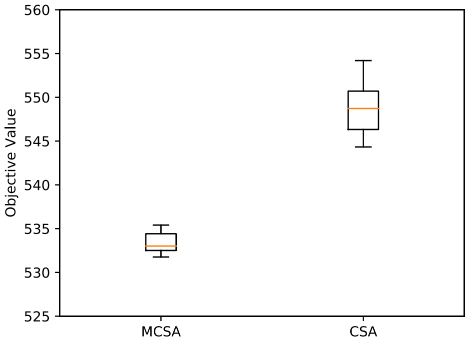
Figure 5. Box plots for comparing 10-runs results of MCSA and CSA on problem instances.



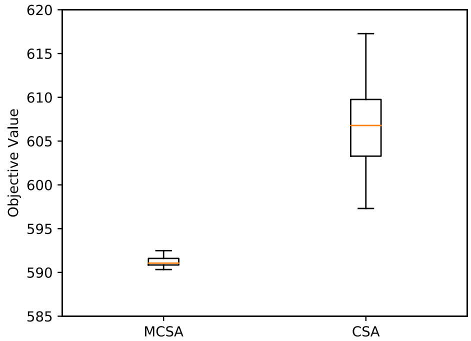
(g) Box plot for WTA7.



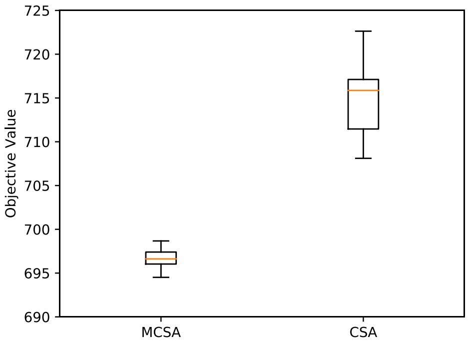
(h) Box plot for WTA8.



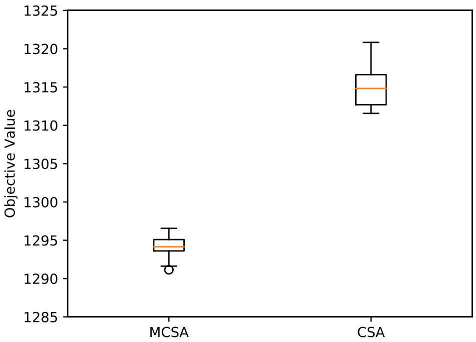
(i) Box plot for WTA9.



(j) Box plot for WTA10.



(k) Box plot for WTA11.



(l) Box plot for WTA12.

Figure 5 (cont). Box plots for comparing 10-runs results of MCSA and CSA on problem instances.

5.2. Comparison of MCSA with the state-of-the-art algorithms

MCSA was compared with four other metaheuristic algorithms for solving the WTA, which are ABC [14], ABC-SA [15], SA [12] and pure CSA. All parameters for the algorithms are given in Table 3. *LIMIT* parameter for MCSA is selected depending on problem size (see in Table 3) as suggested in [24]. With this tuning, *LIMIT* increases when the size of the WTA problem is increased.

The results of all metaheuristic algorithms are compared in terms of the best, mean, worst, median, standard deviation (SD) and time (seconds) in Table 4. However, median and SD values are not available for ABC and ABC-SA. The best results for each problem are shown in bold. Overall, MCSA obtained better results compared to other methods for 11 out of 12 problem instances. All algorithms can achieve the same best results for WTA1 and WTA2. The best result is the same on WTA3 and WTA4 for all algorithms except for CSA. Comparing the results obtained by all metaheuristic algorithms it can be inferred that all algorithms except CSA are successful in reaching the optimum of small size problems.

Table 3. Parameter settings for all algorithms.

ABC [13]		ABC-SA [14]		CSA		MCSA		SA [11]	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
Iteration	200000	Iteration	200000	Iteration	200000	Iteration	200000	Initial Temperature	1000
Population Size	50	Population Size	50	Population Size	40	Population Size	40	Final Temperature	0.1
LIMIT	1000	LIMIT	1000	AP	0.2	AP	0.2	Cooling factor	0.99999
		Initial Temperature	N/A	FL	2	FL	2		
		Final Temperature	N/A			LIMIT	10 x Problem Size		
		Cooling factor	N/A						

Table 4 also shows that the worst value achieved by MCSA is better than the best values achieved by ABC, ABC-SA and CSA for WTA5 to WTA11, which means MCSA provides not only a good exploration but also a good exploitation. According to the results, pure CSA is not efficient yet to solve the WTA problem even if the problem size is small. SD of MCSA is lower than the pure CSA, which indicates that MCSA is a robust algorithm to solve the WTA. For WTA12, ABC-SA achieved the best result comparing to the other algorithms. MCSA is 0.25% worse than ABC-SA for WTA12 according to the best results.

Table 4. Comparison with the state-of-the-art algorithms on the problem instances.

Instance	Weapon	Target	Algorithm	Best	Mean	Worst	Median	SD	Time(sec)
WTA1	5	5	ABC [14]	**48.3640**	48.3640	48.3640	-	-	390.00
			ABC-SA [15]	**48.3640**	48.3640	48.3640	-	-	18.00
			CSA	**48.3640**	48.3640	48.3640	48.3640	0.00	5.20
			MCSA	**48.3640**	48.3640	48.3640	48.3640	0.00	4.42
			SA [12]	**48.3640**	48.3640	48.3640	48.3640	0.00	2985.92
WTA2	10	10	ABC [14]	**96.3123**	96.3123	96.3123	-	-	417.00
			ABC-SA [15]	**96.3123**	96.3123	96.3123	-	-	21.00
			CSA	**96.3123**	96.3123	96.3123	96.3123	0.00	7.10
			MCSA	**96.3123**	96.3123	96.3123	96.3123	0.00	5.39
			SA [12]	**96.3123**	96.3123	96.3123	96.3123	0.00	2841.04
WTA3	20	20	ABC [14]	**142.1070**	142.2480	142.8119	-	-	473.00
			ABC-SA [15]	**142.1070**	142.1070	142.1070	-	-	25.00
			CSA	**142.1070**	143.2052	145.9337	142.7028	1.15	10.92
			MCSA	**142.1070**	142.1070	142.1070	142.1070	0.00	7.56
			SA [12]	**142.1070**	142.1070	142.1070	142.1070	0.00	2752.49
WTA4	30	30	ABC [14]	**248.0285**	248.6854	249.2224	-	-	532.00
			ABC-SA [15]	**248.0285**	248.1678	248.4222	-	-	32.00
			CSA	249.5552	251.8021	254.8158	251.1550	1.79	14.35
			MCSA	**248.0285**	248.0781	248.3312	248.0285	0.10	9.86
			SA [12]	**248.0285**	248.0285	248.0285	248.0285	0.00	2754.31
WTA5	40	40	ABC [14]	305.8729	306.8570	307.4944	-	-	585.00
			ABC-SA [15]	**305.5016**	306.2735	307.1293	-	-	36.00
			CSA	307.7296	312.7559	317.2676	312.7247	2.79	18.78
			MCSA	**305.5016**	305.6046	305.9203	305.5016	0.15	12.70
			SA [12]	**305.5016**	305.5016	305.5016	305.5016	0.00	2760.78
WTA6	50	50	ABC [14]	353.3794	355.1488	356.8539	-	-	654.00
			ABC-SA [15]	353.0149	354.6901	357.2952	-	-	42.00
			CSA	356.7682	361.8349	367.1764	362.0425	3.05	22.60
			MCSA	**353.0102**	353.4104	353.6899	353.4893	0.26	14.86
			SA [12]	353.0767	353.3112	353.5702	353.2610	0.14	2790.03
WTA7	60	60	ABC [14]	414.4555	417.0145	420.1622	-	-	712.00
			ABC-SA [15]	414.7521	417.3107	420.6054	-	-	46.00
			CSA	421.2284	425.7957	429.5839	425.6336	2.09	26.38
			MCSA	**414.2222**	415.4017	416.8135	415.3838	0.82	17.48
			SA [12]	415.0528	415.4068	415.7079	415.4371	0.21	2787.45

Instance	Weapon	Target	Algorithm	Best	Mean	Worst	Median	SD	Time(sec)
WTA8	70	70	ABC [14]	498.0948	500.5102	504.3466	-	-	786.00
			ABC-SA [15]	496.9645	498.3417	500.6414	-	-	52.00
			CSA	508.5992	514.6464	519.7359	515.6737	3.67	30.24
			MCSA	**496.3095**	497.1012	498.1227	497.1297	0.55	19.84
			SA [12]	498.1049	498.5918	499.0167	498.5860	0.30	2841.02
WTA9	80	80	ABC [14]	534.4742	536.8911	541.8093	-	-	831.00
			ABC-SA [15]	531.4078	534.4042	536.5087	-	-	60.00
			CSA	544.3289	548.6797	554.1954	548.7232	2.88	33.99
			MCSA	**531.1592**	533.2647	536.3640	532.9782	1.46	22.26
			SA [12]	534.4408	535.4559	536.2618	535.5937	0.57	2868.79
WTA10	90	90	ABC [14]	592.9167	594.9403	598.3802	-	-	889.00
			ABC-SA [15]	590.4780	592.4761	595.1910	-	-	71.00
			CSA	597.3041	606.4188	617.2749	606.7811	5.52	37.88
			MCSA	**589.3209**	592.5042	594.5376	592.3725	1.52	24.37
			SA [12]	594.0639	595.3277	596.1228	595.6466	0.72	2812.57
WTA11	100	100	ABC [14]	698.4465	701.4467	707.7392	-	-	954.00
			ABC-SA [15]	694.8067	696.3017	700.4310	-	-	79.00
			CSA	708.1073	714.8838	722.6326	715.8635	4.41	41.60
			MCSA	**694.5009**	696.7299	698.3746	696.7235	1.34	29.08
			SA [12]	699.8357	701.0054	702.1189	701.2495	0.75	2805.83
WTA12	200	200	ABC [14]	1295.3142	1299.2044	1303.1223	-	-	1624.00
			ABC-SA [15]	**1287.0240**	1289.1600	1291.2790	-	-	124.00
			CSA	1311.5617	1314.9700	1320.8271	1314.8187	2.74	83.11
			MCSA	1290.2712	1294.4943	1296.3025	1294.8583	1.66	55.72
			SA [12]	1306.9126	1308.3382	1309.4616	1308.5187	0.86	2902.15

A comparison between MCSA and ABC-SA based on time is presented in Figure 6. Although it is not fair to compare MCSA and ABC-SA as we don't know some parameters and number of function evaluations, the capabilities of the used devices for running these two algorithms are approximately similar. It can be shown that the average run time for MCSA is better than ABC-SA.

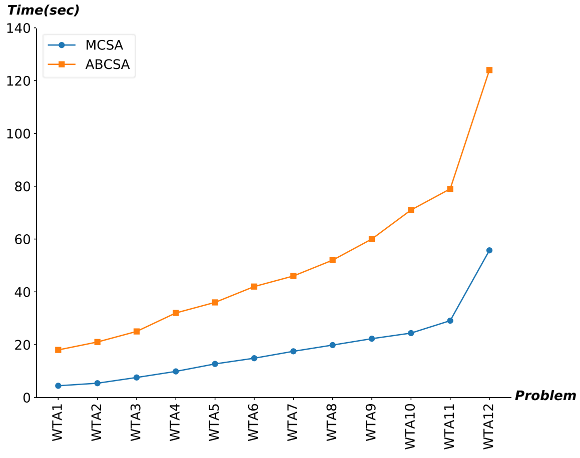


Figure 6. Time comparison between MCSA and ABC-SA for the WTA problem instances.

6. Conclusion and future works

This paper proposed a Modified Crow Search Algorithm (MCSA) for solving the static WTA problem. In MCSA, a trial mechanism that starts with a new position in the search space after a predetermined number of trials, has been adapted to the exploration phase. The number of trials defines as a parameter called *LIMIT*, is adjusted to the size of the problem. With this update, the exploitation stage of CSA is strengthened for combinatorial problems like the WTA. Experimental results of MCSA have been compared with four state-of-the-art algorithms on the WTA problem instances with different dimensions. In each problem, the numbers of the weapons and targets are equal and limited and this limitation occurs the size of the problem. According to the experimental results, MCSA achieved the best results on all problem instances except for only one and outperformed the state-of-the-art algorithms. In future works, MCSA can be combined with single solution based algorithms (hill-climbing, tabu search, simulated annealing, etc.), especially for the second state of CSA. Also, MCSA can be applied to solve dynamic WTA problem or other discrete optimization problems.

References

[1] Kline, A., Ahner, D., & Hill, R. (2018). The Weapon-Target Assignment Problem. *Computers & Operations Research*. https://doi.org/10.1016/j.cor.2018.10.015

[2] Ahuja, R. K., Kumar, A., Jha, K. C., & Orlin, J. B. (2007). Exact and Heuristic Algorithms for the Weapon-Target Assignment Problem. *Operations Research*, *55*(6), 1136–1146. https://doi.org/10.1287/opre.1070.0440

[3] Sikanen, T. (2008). Solving weapon target assignment problem with dynamic programming. *Independent Research Projects in Applied Mathematics*, 32.

[4] Ma, F., Ni, M., Gao, B., & Yu, Z. (2015). An efficient algorithm for the weapon target assignment problem. In *2015 IEEE International Conference on Information and Automation* (pp. 2093–2097). https://doi.org/10.1109/ICInfA.2015.7279633

[5] Lloyd, S. P., & Witsenhausen, H. S. (1986). Weapons allocation is NP-complete. In *1986 Summer Computer Simulation Conference* (pp. 1054–1058).

[6] Talbi, E.-G. (2009). *Metaheuristics: From Design to Implementation*. John Wiley & Sons.

[7] Sotoudeh-Anvari, A., & Hafezalkotob, A. (2018). A bibliography of metaheuristics-review from 2009 to 2015. *International Journal of Knowledge Based Intelligent Engineering Systems*, *22*(1), 83–95. https://doi.org/10.3233/KES-180376

[8] Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Computers & Structures*, *169*, 1–12. https://doi.org/10.1016/j.compstruc.2016.03.001

[9] Şahin, M. A., & Leblebicioğlu, K. (2011). A Hierarchical Fuzzy Decision Maker for the Weapon Target Assignment. *IFAC Proceedings Volumes*, *44*(1), 8993–8998. https://doi.org/10.3182/20110828-6-IT-1002.00986

[10] Wang, J., Luo, P., Hu, X., & Zhang, X. (2018). A Hybrid Discrete Grey Wolf Optimizer to Solve Weapon Target Assignment Problems. *Discrete Dynamics in Nature and Society*. https://doi.org/10.1155/2018/4674920

[11] Li, Y., Kou, Y., Li, Z., Xu, A., & Chang, Y. (2017). A Modified Pareto Ant Colony Optimization Approach to Solve Biobjective Weapon-Target Assignment Problem. *International Journal of Aerospace Engineering*. https://doi.org/10.1155/2017/1746124

[12] Sonuc, E., Sen, B., & Bayir, S. (2017). A Parallel Simulated Annealing Algorithm for Weapon-Target Assignment Problem. *International Journal of Advanced Computer Science and Applications*, *8*(4). https://doi.org/10.14569/IJACSA.2017.080412

[13] Zhang, J., Wang, X., Xu, C., & Yuan, D. (2012). ACGA Algorithm of Solving Weapon—Target Assignment Problem. *Open Journal of Applied Sciences*, *02*(04), 74–77. https://doi.org/10.4236/ojapps.2012.24B018

[14] Durgut, R., Kutucu, H., & Akleylek, S. (2017). An Artificial Bee Colony Algorithm for Solving the Weapon Target Assignment Problem. In *Proceedings of the 7th International Conference on Information Communication and Management* (pp. 28–31). New York, NY, USA: ACM. https://doi.org/10.1145/3134383.3134390

[15] Kutucu, H., & Durgut, R. (2018). Silah Hedef Atama Problemi için Tavlama Benzetimli Bir Hibrit Yapay Arı Kolonisi Algoritması. *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, *22*(Özel), 263. https://doi.org/10.19113/sdufbed.39561

[16] Lee, Z.-J., Lee, C.-Y., & Su, S.-F. (2002). An immunity-based ant colony optimization algorithm for solving weapon–target assignment problem. *Applied Soft Computing*, *2*(1), 39–47. https://doi.org/10.1016/S1568-4946(02)00027-3

[17] Hu, X., Luo, P., Zhang, X., & Wang, J. (2018). Improved Ant Colony Optimization for Weapon-Target Assignment. *Mathematical Problems in Engineering*. https://doi.org/10.1155/2018/6481635

[18] Tokgöz, A., & Bulkan, S. (2013). Weapon Target Assignment with Combinatorial Optimization Techniques. *International Journal of Advanced Research in Artificial Intelligence*, *2*(7). https://doi.org/10.14569/IJARAI.2013.020707

[19] Li, X., Zhou, D., Pan, Q., Tang, Y., & Huang, J. (2018). Weapon-Target Assignment Problem by Multiobjective Evolutionary Algorithm Based on Decomposition. *Complexity*. https://doi.org/10.1155/2018/8623051

[20] Kline, A. G., Ahner, D. K., & Lunday, B. J. (2018). Real-time heuristic algorithms for the static weapon target assignment problem. *Journal of Heuristics*. https://doi.org/10.1007/s10732-018-9401-1

[21] Hocaoğlu, M. F. (2019). Weapon target assignment optimization for land based multi-air defense systems: A goal programming approach. *Computers & Industrial Engineering*, *128*, 681–689. https://doi.org/10.1016/j.cie.2019.01.015

[22] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, *39*(3), 459–471. https://doi.org/10.1007/s10898-007-9149-x

[23] Akay, B. B., & Karaboga, D. (2017). Artificial bee colony algorithm variants on constrained optimization. *An International Journal of Optimization and Control: Theories & Applications (IJOCTA)*, *7*(1), 98–111. https://doi.org/10.11121/ijocta.01.2017.00342

[24] Sonuç, E. (2018). Artificial Bee Colony Algorithm for The Linear Ordering Problem. In *Proceeding Book of the International Conference on Advanced Technologi es, Computer Engineering and Science (ICATCES 2018)* (pp. 818–820). Safranbolu, Turkey.

Emrullah Sonuç *is currently working as an assistant professor at Department of Computer Engineering, Karabuk University, Karabuk, Turkey. He received M.Sc. and Ph.D. degrees from the Department of Computer Engineering, Karabuk University in 2012 and 2017, respectively. His current research interests are: Parallel and distributed computing, Evolutionary computation and Metaheuristic algorithms.*

http://orcid.org/0000-0001-7425-6963