UDK: 001.26,62-11:519.719 "REPTIL"
COSATI: 13-08, 12-05
doi: 10.5937/str2201022M

Mechanical Engineering Design Optimization Using Reptile Search Algorithm

Branislav Milenković [1)]
Đorđe Jovanović [2)]
Mladen Krstić [3)]

Optimization algorithms plays a vital role in mechanical engineering. In this paper we have demonstrated how the Reptile Search Algorithm (RSA) algorithm is able to solve classical engineering design problems. In the first part, the biological reference, as well as a detailed overview of the algorithm is given. Afterwards, the RSA algorithm and the potential to solve the machine engineering design class of problems is given. The source code for this algorithm was written using MATLAB R2020a software suite. The Reptile Search Algorithm (RSA) algorithm was used for optimization problems in the field of engineering design, such as: pressure vessel optimization, disk brake optimization and cantilever beam optimization. The statistical results and comparisons show that the RSA algorithm provides comparable results to other state-of-the-art algorithms used for this problem.

Key words: **reptile, algorithm, engineering.**

Introduction

IN the research field of computer science, there exists a class of problems called NP-problems. The main characteristic of such problems is that the solution to such a problem can be verified within polynomial time complexity, while whether or not such a problem has a solution is unknown. This implies that finding a solution for such a problem is memory and time consuming in most cases.

Optimization problems fall into this category of problems. In this case, the problem is defined using the mathematical problem formulation, where one or more functions are to be minimized or maximized (this value is called the fitness value) under a given set of conditions and the set of variables [18]. Yet the exact optimal value for the fitness value is not known, therefore classifying optimization problems as NP.

One of the most popular methods for solving optimization problems are metaheuristics. What is meant by the term is a class of algorithms having a randomized approach to exploring the vast solution spaces, whose implementation heavily depends upon the problem that is to be solved. Metaheuristics can be split into two categories: single solution (s-type) and population based (p-type). In s-type metaheuristics, a single solution is modified in order to explore the solution space and solve the problem. The main advantage of s-type metaheuristics is that they are memory efficient and allow more creativity in creating new implementations. Yet, careful considerations should be taken while constructing such algorithms, so that the solution space be explored in the most efficient manner. P-type metaheuristics, on the other hand, uses a population of solutions which converges to the best solution of the population. Ofttimes, the algorithm is split in exploration phase, where the search space is explored, and the exploitation phase, where the population converges to the best solution of the population. The advantages to such algorithms are that the solution space is searched thoroughly in a systematic manner. However, these algorithms use more memory than the s-type metaheuristics, and there is not much space in carefully catering the algorithm to the problem that is solved.

To the best of our knowledge, there have not been attempts to solve mechanical design problems using s-type metaheuristics. Therefore, in this section, a brief overview of state-of-the-art p-type algorithms will be presented. In all of these algorithms, what is considered the prey is the current best solution, while all the other search agents are named after the species the algorithm draws inspiration from, unless stated otherwise.

The Honey Badger Algorithm (HBA) [1] is a p-type metaheuristic algorithm which draws inspiration from the behavior of the eponymous mammal species. In this algorithm, there is only one phase, where the constant called the density factor commands the algorithm's convergence. The movement of the badgers can either be in the cardoid shape (called the digging phase) or the badgers can "follow the honey guide bird" (called the honey phase). Which of these two types of movement is applied to the badgers is determined by a random number variable drawn from the uniform distribution.

The Harris' Hawks Algorithm (HHO) [2] draws its inspiration from an American bird of prey which hunts in flocks. In the case of this algorithm, the prey is called the rabbit. This algorithm has both phases of p-type metaheuristic algorithms. The energy parameter is what gradually decreases

1) Faculty of applied sciences, Dušana Popovića 22a, 18000 Niš, SERBIA
2) Mathematical Institute of SASA, Kneza Mihaila 36, 11000 Belgrade, SERBIA
3) Faculty of Mechanical and Civil Engineering, Dositejeva 19, 36000 Kraljevo, SERBIA
Correspondence to: Branislav Milenković, e-mail: bmilenkovic92@gmail.com

with each iteration, and is what determines the transition from exploration to exploitation phase. In the exploration phase, the movement is predetermined, while in the exploitation phase, the hawks have 4 possible strategies for catching the rabbit, dependent upon current energy level and a random variable drawn from the uniform distribution.

The Grey Wolf Optimizer (GWO) [3] is based on the complex hierarchy of grey wolf packs. This algorithm has both phases of p-type metaheuristic algorithms. In the case of exploration phase, all wolves move towards the prey. The main characteristic of this algorithm is how it divides the population in its exploitation phase. Namely, the first, second, and third best solution are named alpha, beta, and delta wolf, while all the other search agents are called omega wolves. In the exploitation phase, the omega wolves move towards the position that is determined by mean of positions of alpha, beta and delta wolf.

The Marine Predator Algorithm (MPA) [4] is based upon the behavior of predator species in the seas and oceans. The main idea is that the predator population chases the prey population. This algorithm is split into three phases instead of the typical two, based three equal parts of number of iterations. The main difference is in constructing the initial solution, which divides the population into predator and prey. First, solutions are randomized to create the initial population of prey, used to determine the movement of the population. Then, the best solution is repeated to form the population of the elite, which represents the predator population. In each phase, the predator population moves faster with each iteration, while the prey population slows down its movement.

The Dingo Optimization Algorithm (DOA) [5] is based upon hunting behavior of dingoes. This algorithm is similar to the GWO algorithm in terms of phases and population division. The first difference is that the encircling phase guarantees that the prey will be surrounded from each part of the solution space. The second difference is that in the exploitation phase, the best, second best, and other search agents' movement happens independent of each other.

The Whale Optimization Algorithm (WOA) [6] is based upon the hunting behavior of humpback whales. There are two phases to this algorithm, but under certain range of values of a random variable, the whales go into encircling the prey. During the exploration phase, the whales move towards a random solution, while in the exploitation phase, the whales move towards the prey in a spiral-like fashion.

The Snake Optimization Algorithm (SOA) [7] is based upon the hunting behavior of snakes. There are the two usual phases of this algorithm, with an addition that there are additional ways of population movement in each phase. In this algorithm, the population is split into two equal parts, which represent the male and female part of the population. Snakes search for food, which represents the value which is increased as iterations pass, and represents the transition from exploration to exploitation. In addition to the movement in the exploration/exploitation phase, there is a chance that the snakes will fight or mate, which explores the solution space further than other algorithms.

The Tunicate Swarm Algorithm (TSA) [8] is inspired by small sea creatures, the tunicates, and their swarm behavior. This algorithm features only one phase, where the movement is decreased as iterations pass. The main characteristic of this algorithm is that, during movement, conflicts among the search agents are resolved, so that the search agents do not end up in the same space, and the tunicates move first towards the best neighbor, and, ultimately, towards the best search agent.

The Grasshopper Optimization Algorithm (GOA) [9] was inspired by the swarm behavior of grasshoppers. Unlike other algorithms, there is only one phase, and all of the population members are treated equally. The factors that affect movement of the search agents are: social forces, dependent upon the distances between a particular grasshopper and all the other grasshoppers, and two constant forces called gravity and wind advection. The social forces are used to bring the algorithm to convergence, reducing the strength of these forces with each iteration.

In this paper, mechanical design problems are solved using the new, state-of-the-art Reptile Search Algorithm (RSA) [10]. The problems that were solved are: pressure vessel, disk brake and cantilever beam.

Reptile search algorithm

The Reptile Search Algorithm (RSA) [10] is based upon the hunting behavior of crocodiles. In nature, the prey is usually encircled and hunted by predators. This behavior in nature was used to inspire such algorithms, where essentially the encirclement phase of the hunt represents a random walk, that is used to search the solution space, while the very hunt represents the exploitation phase, thus bringing the algorithm to convergence.

The core for the inspiration of this algorithm are the manifold advantages of the predatory crocodile species. First, the crocodiles exhibit an anatomical advantage, where their body shape lowers the air and water resistance, enabling them fast movement. Second, crocodiles have advanced cognitive skills in recognizing the behavioral patterns of prey, and, consequently, exploiting their weaknesses in order to capture the prey. Finally, crocodiles hunt in herds, and each member of the herd carries out the same role in every hunt.

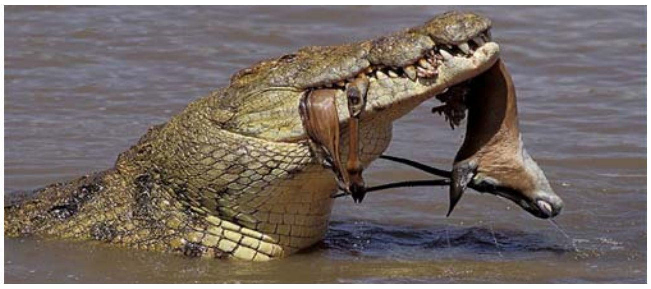
Figure 1. Crocodile hunting waterbuck

This algorithm is divided into two phases: encircling (exploration) and hunting (exploitation). First, the initial solutions are generated at random, using Eq. (1)

$$x_{ij} = rand \times (UB - LB) + LB, j = 1, 2, ..., n \tag{1}$$

Where xij is the j-th dimension of the i-th solution of the population, rand is a random variable, UB and LB are upper and lower bounds, respectively.

After the initial solution is created, the crocodiles go into exploration phase. In this phase, the crocodiles go into the high walking movement strategy and the belly walking movement strategy, each taking up a half of the exploration phase. The movement of crocodiles in this phase is given by Eqs. (2) – (7).

$$x_{(i,j)}(t+1)=\begin{cases} Best_j(t)\times -\eta_{(i,j)}(t)\times\beta - R_{(i,j)}(t)\times rand, & t\le \frac{T}{4} \\ Best_j(t)\times x_{(r_1,j)}\times ES(t)\times rand, & t\le 2\frac{T}{4} \text{ and } t>\frac{T}{4} \end{cases} \quad (2)$$

$$\eta_{(i,j)} = Best_j(t)\times P_{(i,j)}, \quad (3)$$

$$R_{(i,j)} = \frac{Best_j(t)-x_{(r_2,j)}}{Best_j(t)+\varepsilon}, \quad (4)$$

$$ES(t)=2\times r_3\times\left(1-\frac{1}{T}\right), \quad (5)$$

$$P_{(i,j)}=\alpha+\frac{x_{(i,j)}-M(x_i)}{Best_j(t)\times\left(UB_{(j)}-LB_{(j)}\right)+\varepsilon} \quad (6)$$

$$M(x_i)=\frac{1}{n}\sum_{j=1}^{n}x_{(i,j)}, \quad (7)$$

Best$_j$(t) represents the j-th dimension of the best solution, $\eta_{(i,j)}$ the hunting operator for the j-th position in the i-th solution, β the sensitivity parameter equal to 0.1, $R_{(i,j)}$ the reduce function, ES(t) the evolutionary sense function, $P_{(i,j)}$ the percentage difference between the j-th dimension of the i-th solution and the best solution, and M(x_i) is the average position of the i-th solution. The hunting operator is used to control the exploration accuracy, the reduce function is used to reduce the search area, while the evolutionary sense adds to the randomness of the search. The variables denoted by r_i (i being an integer) and ε represent random variables.

The hunting phase represents the latter half of the algorithm. In this phase there are also two movement strategies, which divide this phase into two equal parts, as it is the case in exploration phase. The movement in this phase is given by Eq. (8).

$$x_{(i,j)}(t+1)=\begin{cases} Best_j(t)\times P_{(i,j)}(t)\times rand, & t\le 3\frac{T}{4} \text{ and } t\le 2\frac{T}{4} \\ Best_j(t)-\eta_{(i,j)}(t)\times\varepsilon - R_{(i,j)}(t)\times rand, & t\le T \text{ and } t>3\frac{T}{4} \end{cases} \quad (8)$$

The main characteristic of this algorithm, which makes it stand out from the rest, is the splitting of each of the two phases into two equal sub-phases, thus varying search agent movement.

Optimization models

This section is dedicated to a detailed look into the engineering design problems that are being solved. For each problem, its basis, goal function, input variables, and condition are given. The code of the optimization algorithm was written in the Matlab R2020a software suite.

Optimization of pressure vessel

This problem corresponds to the weight minimization of a cylindrical pressure vessel (Fig.2) with two spherical heads. There are four design variables: the thickness of the pressure vessel (x_1), the thickness of the head (x_2), the inner radius of the vessel (x_3) and the length of the cylindrical component (x_4).

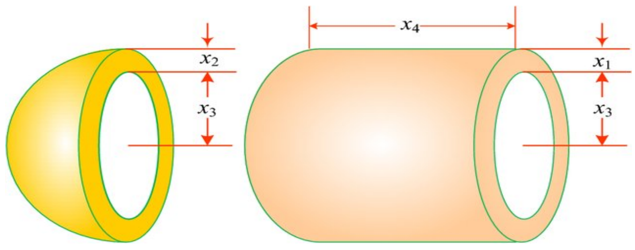

Figure 2. Pressure vessel design problem

The objective functions and constraints of the pressure vessel design optimization are defined as follows:

$$f(X)=0.6224x_1x_3x_4+1.7781x_2x_3^2+ \\ +3.1661x_1^2x_4+19.84x_1^2x_3 \quad (9)$$

$$g_1(X)=-x_1+0.0193x_3\le 0 \quad (10)$$

$$g_2(X)=-x_2+0.00954x_3\le 0 \quad (11)$$

$$g_3(X)=-\pi x_3^2x_4-\frac{4}{3}\pi x_3^3+1296000\le 0 \quad (12)$$

$$g_4(X)=x_4-240\le 0 \quad (13)$$

$$0\le x_i\le 100;\quad i=1,2; \quad (14)$$

$$10\le x_i\le 200;\quad i=3,4; \quad (15)$$

Optimization of disk brake

The main objective of the disk brake optimization problem is minimizing the brake's mass. There are four variables that constitute the input to the metaheuristic algorithm for this problem (as shown in Fig.3): inner radius of the discs (x_1), outer radius of the discs (x_2), engaging force (x_3) and number of the friction surfaces (x_4).

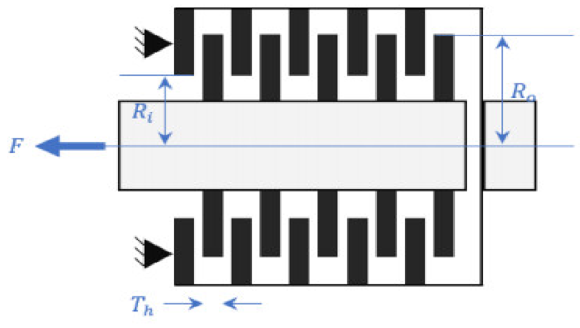

Figure 3. Disk brake design problem

Below are defined the optimization function, as well as problem constraints for the disk brake optimization problem:

$$f_1(x) = 4.9\times10^{-5}\left(x_2^2 - x_1^2\right)\left(x_4 - 1\right), \qquad (16)$$

$$g_1(x) = (x_2 - x_1) - 20 \geq 0. \qquad (17)$$

$$g_2(x) = 30 - 2.5(x_4 + 1) \geq 0. \qquad (18)$$

$$g_3(x) = 0.4 - \frac{x_3}{3.14\left(x_2^2 - x_1^2\right)} \geq 0. \qquad (19)$$

$$g_4(x) = 1 - \frac{2.22\times10^{-3} x_3\left(x_2^3 - x_1^3\right)}{\left(x_2^2 - x_1^2\right)^2} \geq 0. \qquad (20)$$

$$g_5(x) = \frac{2.66\times10^{-2} x_3 x_4\left(x_2^3 - x_1^3\right)}{\left(x_2^2 - x_1^2\right)} - 900 \geq 0. \qquad (21)$$

$$\begin{aligned} 55 &\leq x_1 \leq 80, \\ 75 &\leq x_2 \leq 110, \\ 1000 &\leq x_3 \leq 3000, \\ 2 &\leq x_4 \leq 20. \end{aligned} \qquad (22)$$

Optimization of cantilever beam

Cantilever beam (Fig.4) is an important element in mechanical engineering, whose design is to be handled with utmost care. Minimization of the said beam's weight represents the main goal in design. The lengths of the five bearings are this problem's variables.

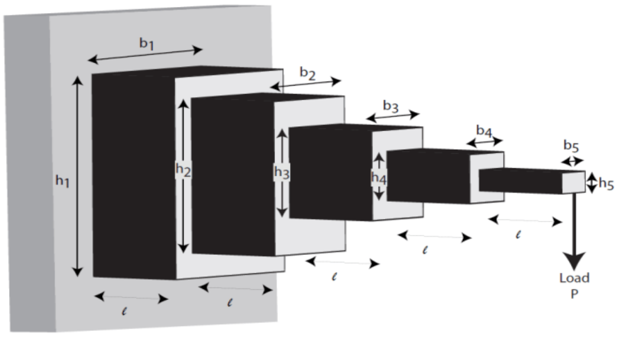

Figure 4. Cantilever beam design problem

This problem's constraints are described in Eqs. (23) to (24) :

$$f(x) = 0,6224(x_1 + x_2 + x_3 + x_4 + x_5), \qquad (23)$$

$$g(x) = \frac{61}{x_1^3} + \frac{27}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0 \qquad (24)$$

The considered variable ranges are described in Eq. (25).

$$0,01 \leq x_1, x_2, x_3, x_4, x_5 \leq 100 \qquad (25)$$

Eexperimental analysis

In this section, the results obtained by using RSA on a set of selected engineering problems is given.

For the pressure vessel problem, the expected value for the goal function is 5885.3327, with the results shown in Table 1.

Table 1. Comparison of results for the pressure vessel problem

Variables	MACA[11]	GOA[9]	WCA[12]	**RSA**
x₁	0.822	0.8736	0.7781	**0.798**
x₂	0.406	0.4318	0.3846	**0.393**
x₃	42.602	45.2666	40.3196	**41.255**
x₄	170.484	199.9998	200	**187.369**
f(x)	5964.50	7666.1258	5888.3327	**5826.81**

In the case of this problem, RSA has given better results than those found in literature.

A detailed display of the results obtained by RSA and a comparison with several results obtained by other methods, for the problem of disk brake, are shown in Table 2.

Table 2. Comparison of results for the pressure disk brake problem

Variables	GA[13]	PSA[14]	PAHS[15]	**RSA**
x_1	65.8	62.6	57.95	**55**
x_2	86.1	83.5	78.57	**75**
x_3	2982.4	2920.9	2736.7	**1000**
x_4	10	11	2	**2**
f(x)	1.66	1.79	0.127	**0.127**

In this case, the RSA gives the same result as the PAHS algorithm, while PSA and GA give worse results.

For the cantilever beam design problem, the results shown in Table 3, along with the results obtained by ALO, MMA and GOA methods.

Table 3. Comparison of results for the cantilever beam problem

Variables	ALO[16]	GOA[9]	MMA[17]	**RSA**
x_1	6.018	6.011	6.010	**6.011**
x_2	5.311	5.312	5.300	**5.349**
x_3	4.488	4.483	4.490	**4.476**
x_4	3.497	3.502	3.490	**3.491**
x_5	2.158	2.163	2.150	**2.145**
f(x)	1.339	1.339	1.340	**1.34**

In this case, the RSA gives the same result as the MMA algorithm, while ALO and GOA give better results.

Conclusion

This paper describes using Reptile search algorithm in order to solve metaheuristic problems from the class of engineering design problems. For this algorithm, the input parameters that were decided on are 10 search agents and 500 iterations. Letting the algorithm run for more iterations and use more search agents did not yield higher quality solutions. Since the execution time is important in order to run more tests, this combination of input parameters was chosen.

For the pressure vessel problem, the RSA algorithm performed better than other listed algorithms from the literature. As for the cantilever beam problem and disk brake problem, the results were shown to be near optimal.

In this field of study, there is always room for further improvements, namely in regard of the betterment of the optimization results.

Literatura

[1] F. Hashim, E. Houssein, K. Hussain, M. Mabrouk, W. Atabany, HONEY BADGER ALGORITHM: NEW METAHEURISTIC ALGORITHM FOR SOLVING OPTIMIZATION PROBLEMS, Mathematics and Computers in Simulation 192 (2022), pp. 84-110.

[2] B. Milenković, IMPLEMETATION OF HARRIS HAWKS OPTIMIZATION (HHO) ALGORITHM TO SOLVE ENGINEERING PROBLEMS, Tehnika, 2021, Vol.76, No.4, pp. 439-446, ISSN 0040-2176, DOI: 10.5937/tehnika2104439M.

[3] B. Milenković, M. Krstić, Đ. Jovanović, APPLICATION OF GREY WOLF ALGORITHM FOR SOLVING ENGINEERING OPTIMIZATION PROBLEMS, Tehnika, 2021.,Vol.76, No.1. pp. 50-57, ISSN 0040-2176.

[4] Faramarzi A., Heidarinejad M., Mirjalili S., Gandomi A., MARINE PREDATORS ALGORITHM: A NATURW – INSPIRED METAHEURISTIC, Expert Systems with Applications 152, 113377, 2020.

[5] B. Milenković, Đ. Jovanović, M. Krstić, AN APPLICATION OF DINGO OPTIMIZATION ALGORITHM (DOA) FOR SOLVING CONTINUOUS ENGINEERING PROBLEMS, FME Transactions, 2022, Vol.50, No.2, pp. ISSN 1451-2092 (print), doi:10.5937/fme2201331M

[6] Mirjalili, S.; Lewis, A. THE WHALE OPTIMIZATION ALGORITHM, Adv. Eng. Softw. 2016, 95, pp. 51–67.

[7] F. Hashim, A. Houssein, SNAKE OPTIMIZER: A NOVEL META-HEURISTIC OPTIMIZATION ALGORITHM, Knowledge-Based Systems, 2022, doi:10.1016/j.knosys.2022.108320

[8] M. Krstić, B. Milenković, Đ. Jovanović, APPLICATION OF THE METAHEURISTIC TUNICATE SWARM ALGORITHM IN SOLVING APPLIED MECHANICS PROBLEMS, Journal of Production Engineering (JPE), University of Novi Sad Faculty of Technical Sciences, December 2021, Vol.24, No.2, pp. 31-34, ISSN 1821-4932.

[9] Saremi S., Mirjalili S., Lewis A., GRASSHOPPER OPTIMIZATION ALGORITHM: THEORY AND APPLICATION, Advances in Engineering Software 105, pp. 30-47, 2017.

[10] L. Abualigah, M. Elaziz, P. Sumari, Z. Geem, A. Gandomi, REPTILE SEARCH ALGORITHM (RSA): A NATURE -INSPIRED METAHEURISTIC OPTIMIZER, Expert Systems with Applications 191(11):116158, doi:10.1016/j.eswa.2021.116158

[11] V.Grković, R. Bulatović, MODIFIED ANT COLONY ALGORITHM FOR SOLVING ENGINEERING OPTIMIZATION PROBLEMS, IMK-14-Research and Development 18, 4, 2012.

[12] H. Eskandar, A.Sadollah, A.Bahreininejad, M.Hamdi: WATER CYCLE ALGORITHM-A NOVEL METAHEURISTIC OPTIMIZATION METHOD FOR SOLVING CONSTRAINED ENGINEERING OPTIMIZATION PROBLEMS, Computers and Structures 2012.

[13] J.L.Marcelin, "GENETIC OPTIMISATION OF GEAR," International Journal of Advanced Manufacturing Technology, vol. 17, no. 12, pp. 910–915, 2001.

[14] P.Sabarinath, M. R. Thansekhar, and R. Saravanan, "PERFORMANCE EVALUATION OF PARTICLE SWARM OPTIMIZATION ALGORITHM FOR OPTIMAL DESIGN OF BELT PULLEY SYSTEM, in Swarm, Evolutionary, and Memetic Computing, vol. 8297 of Lecture Notes in Computer Science, pp. 601–616, Springer, Cham, Switzerland, 2013.

[15] P.Sabarinath, M. Thansekhar, R. Saravanan, MULTIOBJECTIVE OPTIMIZATION METHOD BASED ON ADAPTIVE PARAMETER HARMONY SEARCH ALGORITHM, Journal of Applied Mathematics, Volume 2015, Article ID 165601, 12 pages.

[16] Mirjalili S., THE ANT LION OPTIMIZER, Advances in Engineering Software 83, pp. 80-98, 2015.

[17] Chickermane H., Gea H., STRUCTURAL OPTIMIZATION USING A NEW LOCAL APPROXIMATION METHOD, International Journal for Numerical Methods in Engineering, 39:829 46, 1996.

[18] Talbi El-Ghazali, METAHEURISTICS FOR BI-LEVEL OPTIMIZATION, Studies in Computational Intelligence, Springer, 2013.

Received: 10.06.2022.
Accepted: 25.07.2022.

Optimizacija u mašinskom inženjerstvu implementacijom algoritma reptila

Optimizacione metode igraju vrlo značajnu ulogu u mašinskom inženjerstvu. Cilj ovog rada je primena algoritma reptile (Reptile Search Algorithm) u cilju rešavanja klasičnih mašinskih problema u praksi. U drugom poglavlju data je biološka referenca algoritma, kao i njegov deteljan pregled. Nakon toga su prikazani rezultati koji su dobijeni implementacijom algoritma reptila. Izvorni kod ovog algoritma je napisan u Matlab R2020a jeziku. Sam prethodno navedeni algoritam je upotrebljen za rešavanje problema poput: suda pod pritiskom, disk kočnice i konzolne grede. Relevantni rezultati su dobijeni i samo poređenje sa ostalim optimizacionim algoritmima pokazuje efikasnost algoritma reptila.

Kjučne reči: **reptil, optimizacija, inženjerstvo.**