An Educational Tool Aimed at Learning Metaheuristics

Md. Abdul Kader
Faculty of Computing
Universiti Malaysia Pahang
Pahang, Malaysia
kdr2k10@gmail.com

Jamal A. Jamaluddin
Pakaratwork Sdn. Bhd
47820 Petaling Jaya
Selangor, Malaysia
jamal@pakaratwork.com

Kamal Z. Zamli
Faculty of Computing
Universiti Malaysia Pahang
Pahang, Malaysia
kamalz@ump.edu.my

ABSTRACT
In this paper, we introduce an education tool for learning metaheuristic algorithms that allows displaying the convergence speed of the corresponding metaheuristic upon setting/changing the dependable parameters. This tool can be an educational assistant for beginners to learn metaheuristic in theoretical lectures as well as practical sessions. Implemented with Java, this tool provides a friendly GUI for setting the parameters and display the result from where the learner can see how the selected algorithm converges for a particular problem solution. Initially, this tool adopts only Crow Search, Jaya, and Sine Cosine algorithms. But more metaheuristics will be included in future research. However, this application is a useful tool that will help not only beginner learners but also the researchers. This paper also describes the proposed software tool and the mentioned metaheuristics in detail and provides future research work on it.

CCS Concepts
• Computing methodologies → Symbolic and algebraic manipulation → Symbolic and algebraic algorithms → Optimization algorithms

Keywords
Metaheuristic Algorithm; Crow Search Algorithm (CSA); Jaya Algorithm (JA); Sine Cosine Algorithm (SCA); Applications of CSA; Metaheuristic Algorithms Learning Tool.

1. INTRODUCTION
Resource depletion forces people to make maximum profit at minimum cost, which is generally known as optimization. Several methods are proposed to solve optimization problems in literature. Among them, metaheuristic algorithms show enormous interest to the researchers for solving severe real-world optimization problems. These algorithms have global search capability (but no guarantee for getting optimal solution), provide solution rapidly, and offer easy handling and less time consumption to solve any complex real-world problem [1]. Many classification criteria may be used for metaheuristics [2] like nature-inspired metaheuristics versus non-nature-inspired or other metaheuristics, memory usage versus memoryless methods, deterministic versus stochastic, single-solution based search versus population-based search, greedy versus iterative. Among them, this paper emphasizes on the first broad category of classification that is nature-inspired versus non-nature-inspired classification.

ICSCA 2020, February 18–21, 2020, Langkawi, Malaysia

ACM ISBN 978-1-4503-7665-5/20/02…$15.00
https://doi.org/10.1145/3384544.3384597

There are many nature-inspired metaheuristics are proposed in the literature. Crow Search Algorithm (simulate the intelligent food hiding behavior of crows) [3], Sooty Tern Optimization Algorithm (simulate the migration and attacking behaviors of sooty terns in real life) [4], Salp Swarm Algorithm (simulate swarming behavior of salps during navigating and foraging in oceans) [5], Owl Search Algorithm (simulate hunting mechanism of the owls in dark) [6] and Squirrel Search Algorithm (simulate dynamic foraging and gliding behavior) [7], Nomadic People Optimizer (simulate living behavior) [8] are recently developed and most popular metaheuristics. Besides, some example of other (non-nature-inspired) metaheuristics are Jaya Algorithm (always move towards the best solution and should avoid the worst solution) [9], Sine Cosine Algorithm (fluctuate outwards or towards the best solution using sine cosine function) [10], TLBO (simulate the effect of influence of a teacher on learners) [11], Henry Gas Solubility Optimization (simulate the behavior of Henry's law) [12], Simulated Annealing (simulate the annealing technique used in metallurgy) [13], Artificial Electric Field Algorithm (simulates the Coulomb's law of electrostatic force) [14]. Despite having a large variety of metaheuristic algorithms, there are still lack of tools [15, 16] to support learners (beginners or researchers in related areas) with the easiest and cost-effective way.

There exists a few example of this kind of tool in literature such as in [17], a lego robot-based platform for learning metaheuristics through robot experiments which requires a costly hardware platform, in [18], an educational software tool (Problem Metaheuristic Solver) for the generic study of the concepts related to the optimization field which emphasizes on designing and analyzing the behavior of metaheuristics in a complicated way, and in [19], a web-based educational tool where two metaheuristics are adopted for benchmark functions only.

Over the last decades, the evolution of technologies that provides opportunities to create digital learning environments that complement traditional learning systems. This is the main inspiration for the development of the proposed educational software tool that can be used for learning metaheuristic algorithms. For the initial development and simplicity of the proposed software tool, only three algorithms are adopted. These are the Crow Search Algorithm (CSA) from the nature-inspired category and Jaya Algorithm (JA) and Sine Cosine Algorithms (SCA) from non-nature-inspired or another category.

The first algorithm, CSA, offers ease implementation and less adjustable parameters which imitates the intelligent behavior of crows for hiding and finding food. It provides outstanding results when solving engineering design/optimization problems. There are many improvements and hybridizations of CSA have been

proposed in the literature for different optimization problems because no metaheuristics can be useful for all optimization problems. The second algorithm, Jaya which is easier, more efficient, and more powerful algorithm for finding the global best solution. It has been applied to many benchmark functions for constrained and unconstrained problems successfully. The third algorithm, Sine Cosine algorithm which creates random candidate solutions initially and fluctuates them towards and backwards the best solution using sine cosine functions. During optimization, it avoids the local optima and converges to global optima effectively.

The name of this proposed software tool is ETLMA in short which has a graphical user interface that can dynamically display the convergence rate to find the best solution of the corresponding optimization algorithm. Parameter settings of selected metaheuristic can be changed/modified and display the corresponding convergence graph dynamically. It helps the users or learners to understand the effect of changing the parameter settings of the corresponding metaheuristic in the convergence graph.

ETLMA tool is coded in Java, run in windows 10 using CPU Intel core i7 2.20 GHz speed, and 8GB RAM. Learners can experiment using ETLMA as long as they run the ETLMA software. Moreover, ETLMA can be applied for digital educational purposes. Educators can easily show beginners to better understand the convergence rate upon changing the parameter settings of selected metaheuristic.

The paper is structured accordingly as follows. Section 2 presents overview of the original CSA, JA, and SCA algorithms. A detailed discussion of ETLMA is presented in Section 3. Section 4, outlines the applications of CSA in engineering design problems and JA and SCA. Finally, Section 5 concludes this study and gives the scope for future work.

2. OVERVIEW OF METAHEURISTIC ALGORITHMS

2.1 Crow Search Algorithm (CSA)

This sub-section presents a general overview of the standard Crow Search Algorithm (CSA). This algorithm is a stochastic population-based nature-inspired novel metaheuristic algorithm that is proposed by Alireza Askarzadeh in 2016 [3] for solving continuous optimization problems. This algorithm was inspired by the intelligent behavior of Crows for hiding excess food. Each crow searches for better food sources or hiding places in the environment. The working principle of the CSA is described below.

The pseudocode of CSA is shown in Algorithm 1. It contains eight steps that briefly stated as follows:

1. Initialization of optimization problem (objective function, decision variables, and constraints) and CSA parameters (flock length (N), maximum number of iteration (itr_{max}), flight length (fl), awareness probability (AP)).
2. Initialization positions with randomly generated feasible solution vector of the crows and fill the memory (m) by the initial position.
3. Fitness or objective function evaluation (fitness quality is evaluated using the decision variable values).
4. Crows new position generation by the following equation (repeated for all the crows) where $x^{i,itr}$ is the position vector of i^{th} crow in itr iteration.
5. Check and update the crow's location if the position is feasible.
6. Fitness evaluation for the new locations.
7. If the fitness of the new position is better than the fitness of memory position than update memory with the new position.
8. Check to stop criterion (repeated until itr_{max} is reached).

Algorithm 1: Pseudocode of the original CSA

Procedure CSA
  Randomly initialize N crows positions in the search space
  Evaluate the position of the crows
  Memory initialization of each crow
  while $itr < itr_{max}$ **do**
    for i = 1: *N (all N crows of the flock)* **do**
      Randomly choose one of the crows to follow (for example j)
      Define an awareness probability
      Generate $r \in [0, 1]$
      if $r \geq AP$ **then**
        $x^{i,itr+1} = x^{i,itr} + r_i \times fl \times (m^{j,itr} - x^{i,itr})$
      else
        $x^{i,itr+1}$ = a random position
      end if
    end for
    Check the feasibility of new positions
    Evaluate the new position of the crows
    Update the memory of crows
    $itr = itr+1$
  end while
end procedure

2.2 Jaya Algorithm (JA)

This sub-section presents a general overview of the Jaya algorithm. It works by establishing the solution to problems through avoiding the worst solutions and moving towards the best optimal solution. Although it is algorithm-specific parameter-free, its performance depends on only a few control parameters, which are common to many optimization algorithms like population size, number of generations, and number of design variables [20]. The working principle of the Jaya algorithm is described below.

Algorithm 2: Pseudocode of the original JA

Procedure JA
  Randomly initialize population size,
    number of design variables, termination criteria
  while $itr < itr_{max}$ **do**
    Identify best and worst solution
    Modify the solutions based on best and worst solutions
    $X'_{j,k,i} = X_{j,k,i} + r_{1,j,i} (X_{j,best,i} - |X_{j,k,i}|) - r_{2,j,i} (X_{j,worst,i} - |X_{j,k,i}|)$
    if $X'_{j,k,i} > X_{j,k,i}$ **then**
      accept and replace the previous solution
    else
      keep the previous solution
    end if
    $itr = itr+1$
  end while
  Report the optimum solution
end procedure

In Algorithm 2, the Jaya algorithm begins by initializing its basic parameters. These are termination criteria (here the maximum number of iterations are considered as the termination condition), population size (number of candidate solutions), and number of

design variables. In the second and third steps, the best and worst solution is identified from the population and modify the solutions. In the fourth step shown, it is seen that if the updated solution is better than the previous solution, then it accepts and replaces the previous solution otherwise keeps the previous solution. In the final step, if the termination condition is satisfied, then it reports the optimum solution; otherwise, get back to the second step of this algorithm.

2.3 Sine Cosine Algorithm (SCA)

The sine-cosine algorithm (SCA) is a popular population-based metaheuristic algorithm proposed by Mirjalili [10]. It creates random candidate solutions initially and fluctuates them towards and backwards the best solution using sine cosine functions which are the same operator with a 90-degree phase shift [21]. During optimization, it avoids the local optima and converges to global optima effectively. The working principle of the Sine Cosine Algorithm is described below.

Algorithm 3: Pseudocode of the original SCA

Procedure SCA
Randomly initialize a set of search agents (solutions) (X)
while $itr < itr_{max}$ **do**
Evaluate each of the search agents by the objective function
Update the best solution obtained so far (P=X)*
Update r_1, r_2, r_3, *and* r_4
Update the position of the search agents by following equation

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times \sin(r_2) \times |r_3 \times P_i^t - X_i^t|, & r_4 < 0.5 \\ X_i^t + r_1 \times \cos(r_2) \times |r_3 \times P_i^t - X_i^t|, & r_4 \geq 0.5 \end{cases}$$

$itr = itr+1$
end while
Report the global optimum solution
end procedure

In Algorithm 3, the SCA begins by initializing a set of search agents or solutions (X), termination criteria (here the maximum number of iterations are considered as the termination condition). The algorithm then evaluates each search agent by objective function and updates the best solution obtained so far. Then update the random variables $\boldsymbol{r_1}$, $\boldsymbol{r_2}$, $\boldsymbol{r_3}$, *and* $\boldsymbol{r_4}$ which will be used in the equation mentioned in Algorithm 3 to update the position of the search agents. In the final step, if the termination condition is satisfied, then it reports the optimum solution, and the SCA algorithm terminates the optimization process by default.

3. PROPOSED TOOL (ETLMA)

The recommended metaheuristic learning tool (ETLMA) is an educational software tool developed in Java Standard Edition 8 and aimed at learning the generic concept related to the metaheuristic. ETLMA is focused on studying metaheuristics, CSA, JA, and SCA. In this context, ETLMA has been designed to be used as an assistant during the theoretical lectures and as an experimental tool during the practical session, especially for convergence analysis. The main objective of ETLMA is to encompass the stages shown in Figure 1 when solving optimization problems as follows.

1. Problem Selection: ETLMA provides options to choose optimization problems for finding the best solution.
2. Configure the Parameters: The selected metaheuristic has multiple components/parameters that are needed to initialize, which determine how the search space will explore. ETLMA will set the parameters for the selected optimization problems automatically, which are changeable for further analysis.
3. Execution of metaheuristic: Upon completion of the parameter settings, this stage will calculate the convergence rate of the selected metaheuristic for the specific optimization problem.
4. Assessment of Result: ETLMA eases the assessment by generating the convergence curve, which shows that how selected metaheuristic behaves for the specific optimization problem.

Figure 2 portrays a friendly graphical user interface (GUI) of ETLMA. And an example of calculating and showing convergence rate using arbitrary values of selected metaheuristic's parameters for finding the optimal solution is shown in Figure 3.

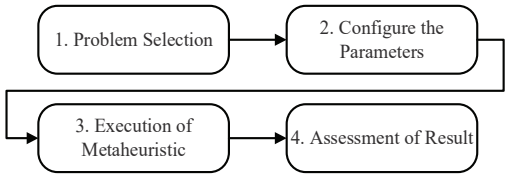

Figure 1. Stages of solving optimization problems by ETLMA.

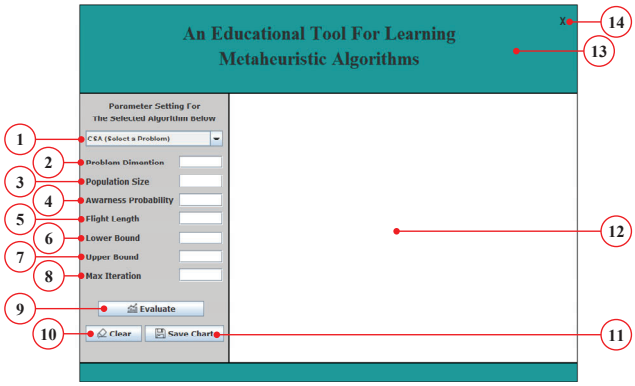

Figure 2. Graphical User Interface of ETLMA Tool.

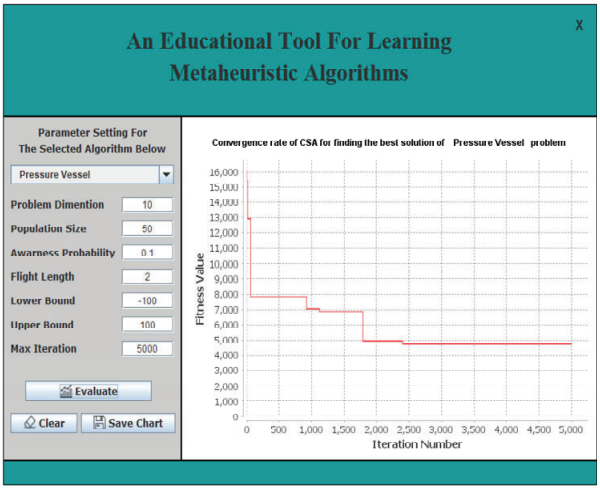

Figure 3. An example of calculating and showing convergence rate using arbitrary values of selected metaheuristic's parameters for finding the optimal solution.

The GUI is organized and labeled in the following parts:

1. Problem Selection: There are three metaheuristics (CSA, JA, and SCA) and five optimization problems that are added in ETLMA, which will be described in the later section.

2. Problem Dimension: Search space dimension. (only for CSA)
3. Population Size: Number of searchers or population size.
4. Awareness Probability (*AP*): Convergence rate will increase for the small values of *AP* and vice versa. (only for CSA)
5. Flight Length (*fl*): *fl*<1 leads the local search and *fl*>1 leads to the global search. (only for CSA)
6. Lower Bound: Minimum value of any random location in search space.
7. Upper Bound: Maximum value of any random location in search space.
8. Max Iteration: Termination criteria.
9. Evaluate: Clicking this button will calculate and show the convergence curve for the selected metaheuristic.
10. Clear: Clicking this button will clear all the parameter values and set them to null.
11. Save Chart: Clicking this button will save the convergence curve shown in part no. 12.
12. Convergence Curve: For each clicking on the button "Evaluate" will show the convergence curve in this area.
13. Displays the name of the proposed software tool.
14. Exit/Closing the ETLMA.

4. APPLICATIONS OF METAHEURISTICS IN ENGINEERING DESIGN PROBLEMS

In ETLMA, six engineering optimization (Design) problems that are mentioned in [3] are used for experimenting the CSA's convergence rate. The parameter settings for solving those design problems are presented in Table 1.

Table 1. Parameter setting used in ETLMA for solving engineering design problems [3].

SI	Engineering Design Problem	*N*	itr_{max}	*fl*	*AP*
1.	Three-Bar Truss	50	500	2	0.1
2.	Pressure Vessel	50	5000	2	0.1
3.	Tension/Compression Spring	50	1000	2	0.1
4.	Welded Beam	50	2000	2	0.1
5.	Gear Train	20	500	2	0.1
6.	Belleville Spring	50	1000	2	0.1

Although there are a few default optimization problems added in ETLMA, this tool can be used for other optimization problems by setting the parameters' value to show the convergence curve. Figure 4-6 shows the CSA's convergence rate for finding the global best solution to the corresponding optimization problem. For the limitation of the number pages, only figures (CSA's convergence rate) for the first three engineering optimization problems mentioned in Table 1 are shown. Figure 7 and Figure 8, show the convergence rate for the selected function [(x-2)*(x-4)].

5. CONCLUSION AND FUTURE WORK

This paper describes an education software tool ETLMA for assisting the learners in understanding metaheuristics (CSA, JA, SCA). This software tool attempts to conduit the gap between the learning of theory and practice by exploring the features of metaheuristics. Using the ETLMA, learners able to see and understand the effect of updating the parameter setting of the selected metaheuristic dynamically. Future research works will include more metaheuristics and benchmark functions in this tool. Additionally, a learner can pick the better metaheuristic which convergences faster than the others for the same parameter settings. Authors of this paper believe that ETLMA is useful for learning metaheuristic fundamentals.

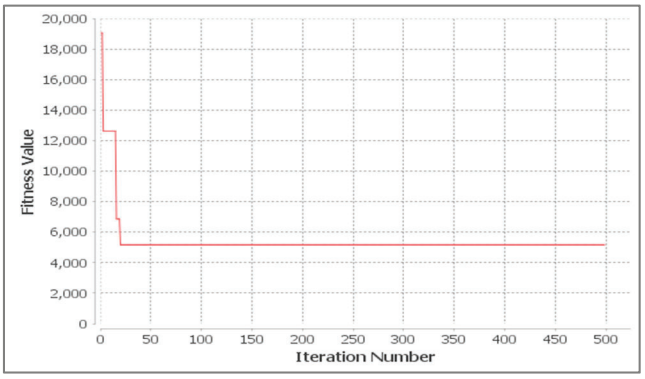

Figure 4. Graph showing the CSA's convergence rate for the Three-Bar Truss problem.

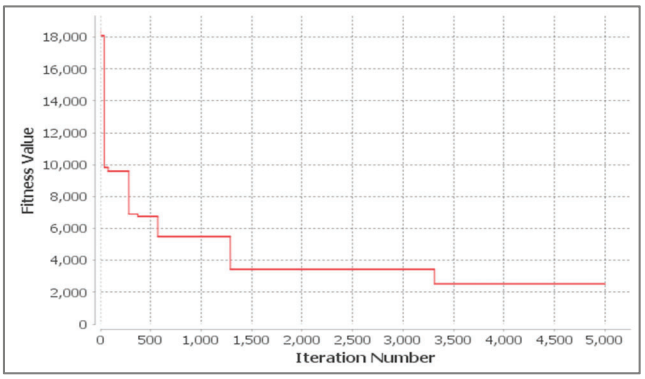

Figure 5. Graph showing the CSA's convergence rate for the Pressure Vessel problem.

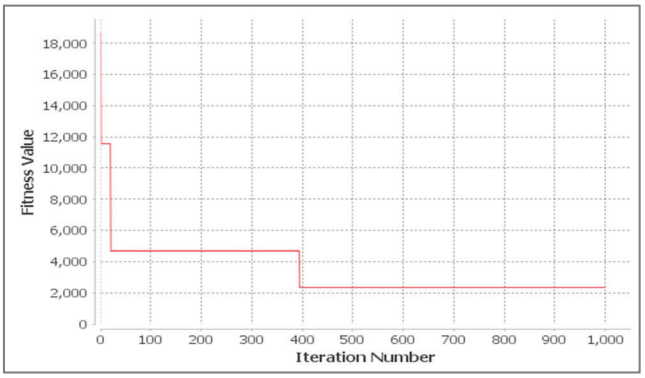

Figure 6. Graph showing the CSA's convergence rate for the Tensor/Compression Spring problem.

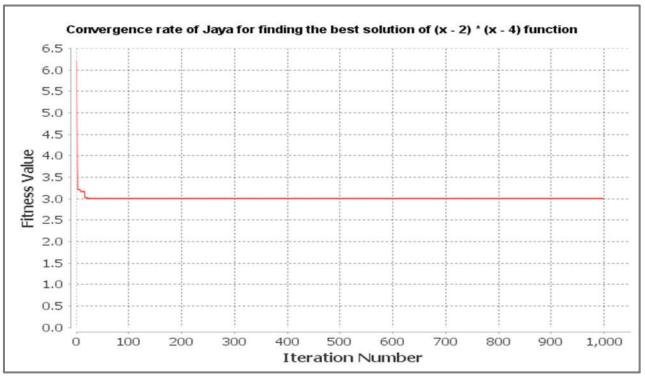

Figure 7. Graph showing the JA's convergence rate for the selected function.

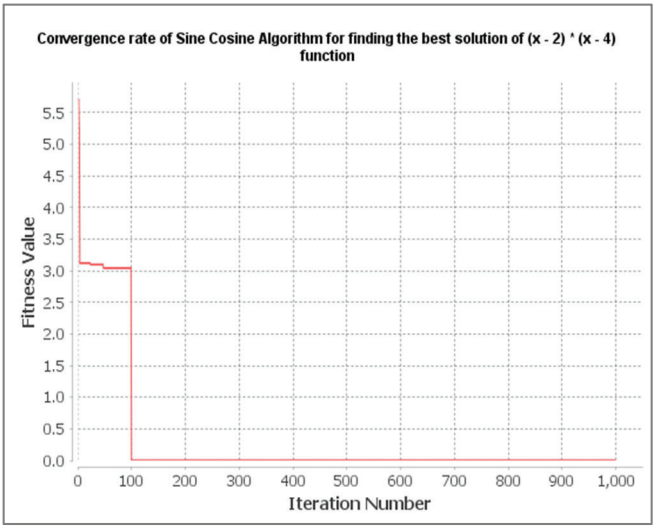

Figure 8. Graph showing the SCA's convergence rate for the selected function.

6. ACKNOWLEDGMENTS

This research is funded by RDU Grant No. UIC191202: The Development of T-Way Test Generation Tool for Combinatorial Testing from Universiti Malaysia Pahang.

7. REFERENCES

[1] M. Jain, A. Rani, and V. Singh, "An improved Crow Search Algorithm for high-dimensional problems," *Journal of Intelligent & Fuzzy Systems,* vol. 33, no. 6, pp. 3597-3614, 2017.

[2] E.-G. Talbi, *Metaheuristics: From Design to Implementation*. Wiley Publishing, 2009, p. 593.

[3] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm," *Computers & Structures,* vol. 169, pp. 1-12, 2016/06/01/ 2016, doi: https://doi.org/10.1016/j.compstruc.2016.03.001.

[4] G. Dhiman and A. Kaur, "STOA: A bio-inspired based optimization algorithm for industrial engineering problems," *Engineering Applications of Artificial Intelligence,* vol. 82, pp. 148-174, 2019/06/01/ 2019, doi: https://doi.org/10.1016/j.engappai.2019.03.021.

[5] S. Mirjalili, A. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. Mirjalili, "Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems," *Advances in Engineering Software,* 07/01 2017, doi: 10.1016/j.advengsoft.2017.07.002.

[6] M. Jain, S. Maurya, A. Rani, and V. Singh, "Owl search algorithm: A novel nature-inspired heuristic paradigm for global optimization," *Journal of Intelligent & Fuzzy Systems,* vol. 34, pp. 1573-1582, 03/22 2018, doi: 10.3233/JIFS-169452.

[7] M. Jain, V. Singh, and A. Rani, "A novel nature-inspired algorithm for optimization: Squirrel search algorithm," *Swarm and Evolutionary Computation,* vol. 44, 02/01 2018, doi: 10.1016/j.swevo.2018.02.013.

[8] S. Q. Salih and A. A. Alsewari, "A new algorithm for normal and large-scale optimization problems: Nomadic People Optimizer," *Neural Computing and Applications,* pp. 1-28, 2019.

[9] R. Venkata Rao, "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *International Journal of Industrial Engineering Computations,* vol. 7, pp. 19-34, 01/01 2016, doi: 10.5267/j.ijiec.2015.8.004.

[10] S. Mirjalili, "SCA: a sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems,* vol. 96, pp. 120-133, 2016.

[11] R. V. Rao, V. Savsani, and J. Balic, "Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems," *Engineering Optimization,* vol. 44, no. 12, pp. 1447-1462, 2012.

[12] F. A. Hashim, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, and S. Mirjalili, "Henry gas solubility optimization: A novel physics-based algorithm," *Future Generation Computer Systems,* vol. 101, pp. 646-667, 2019.

[13] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science,* vol. 220, no. 4598, pp. 671-680, 1983.

[14] A. Yadav, "AEFA: Artificial electric field algorithm for global optimization," *Swarm and Evolutionary Computation,* vol. 48, pp. 93-108, 2019.

[15] K. Z. Zamli, N. A. M. Isa, M. F. J. Klaib, and S. N. Azizan, "A tool for automated test data generation (and execution) based on combinatorial approach," *International Journal of Software Engineering and Its Applications,* vol. 1, no. 1, pp. 19-35, 2007.

[16] A. A. Alsewari and K. Z. Zamli, "Interaction test data generation using harmony search algorithm," in *2011 IEEE Symposium on Industrial Electronics and Applications*, 2011: IEEE, pp. 559-564.

[17] D. Zaldivar, E. Cuevas, O. Maciel, A. Valdivia, E. Chavolla, and D. Oliva, "Learning classical and metaheuristic optimization techniques by using an educational platform based on LEGO robots," *The International Journal of Electrical Engineering & Education,* p. 0020720918822738, 2019.

[18] C. Expósito-Izquierdo, I. López-Plata, and J. M. Moreno-Vega, "Problem MetaHeuristic Solver: An educational tool aimed at studying heuristic optimization methods," *Computer Applications in Engineering Education,* vol. 23, no. 6, pp. 897-909, 2015.

[19] G. Sarıman and E. U. Küçüksille, "Web based educational tool for metaheuristic algorithms," *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi,* vol. 20, no. 2, pp. 46-53, 2014.

[20] A. B. Nasser, F. Hujainah, A. A. Al-Sewari, and K. Z. Zamli, "An Improved Jaya Algorithm-Based Strategy for T-Way Test Suite Generation," Cham, 2020: Springer International Publishing, in Emerging Trends in Intelligent Computing and Informatics, pp. 352-361.

[21] K. Z. Zamli, F. Din, B. S. Ahmed, and M. Bures, "A hybrid Q-learning sine-cosine-based strategy for addressing the combinatorial test suite minimization problem," *PloS one,* vol. 13, no. 5, p. e0195675, 2018.