*Corresponding Author Email: tahirsag@selcuk.edu.tr
Proceedings of the International Conference on Academic Research in Science, Technology and Engineering, Vol. 1, Issue. 1, 2022, pp. 1-11
DOI: https://doi.org/10.33422/icarste.v1i1.12

ISSN: 2783-7742 online

Diamond Scientific Publishing

Approaches Used in Adapting Metaheuristic Optimization Algorithms Developed for Continuous Problems to Discrete Problems

Tahir Sağ
Department of Computer Engineering, Selçuk University, Turkey

Abstract

Many real-world problems such as determining the type and number of wind turbines, facility placement problems, job scheduling problems, are in the category of combinatorial optimization problems in terms of the type of decision variables. However, since many of the evolutionary optimization algorithms are developed for solving continuous optimization problems, they cannot be directly applied to optimization problems with discrete decision variables. Therefore, the continuous decision variable values generated by these metaheuristics need to be converted to binary values using some techniques. In other words, to apply such algorithms to discrete optimization problems, it is necessary to adapt the candidate solution vectors of the algorithms to discrete values and make changes in their working structures. In this study, firstly, adaptation methods that are frequently used in previous studies in transforming metaheuristic optimization algorithms designed for the solution of continuous optimization problems into discrete optimization algorithms are explained. Then, the popular location update strategies used in solving discrete optimization problems are explained. The presented work summarizes the process of adapting continuous optimization algorithms to solve combinatorial problems step by step.

Keywords: metaheuristics, combinatorial optimization, transfer functions, location update strategies

1. Introduction

Optimization problems are frequently encountered in many areas such as production, planning, design, and engineering. To solve these problems, it is aimed to maximize or minimize a cost function created with the decision variables. It is known that metaheuristic algorithms are used successfully on optimization problems that cannot be modelled deterministically or which are difficult to solve in exponential time (NP-Hard). These algorithms do not guarantee to reach the global optimum due to their stochastic operating characteristics, but they often converge iteratively to an acceptable near-optimal solution, avoiding excessive time and resource consumption.

Optimization problems are classified into two groups as continuous and discrete problems according to the values of the decision variables (Yang, 2010). In this context, an optimization problem is a continuous problem if variables can take infinite values in search space. On the other hand, discrete optimization problems take some specific integer values, unlike continuous optimization. For example, traveling salesman problem is defined as a discrete optimization problem, since each city is represented by an integer. Binary optimization is a subpart of discrete optimization. In binary optimization, the decision variables take only the value 0 or 1. In facility placement problems, a value of 0 indicates that the facility will not be placed in the relevant area, and a value of 1 indicates that the facility will be placed. In optimization algorithms, it is important whether the candidate solution generated during the location update stage is within the range of decision variables and whether it is in the feasible region where the constraint functions defined depending on the same decision variables are also satisfied. In combinatorial optimization problems, the candidate solution is based on the permutation of integer values.

Vehicle routing problems, determination of the type of wind turbines, plant placement problems, job scheduling problems, and many other real-world problems are combinatorial optimization problems that are difficult to formulate and solve. However, metaheuristic optimization algorithms in the literature are generally developed for the solution of continuous optimization problems. Continuous optimization algorithms cannot be used directly for the solution of discrete optimization problems. To apply such algorithms to discrete optimization problems, some changes should be made in the working structures of the algorithms. Many studies proposed in this direction are available in the literature.

Kennedy and Eberhart (Kennedy & Eberhart, 1997) modified the Particle Swarm Optimization (PSO) algorithm to operate in binary space and named it BPSO; here the sigmoid function is used as a logistic function to map from continuous-valued variables to binary values. Saha et al. (Saha, Kole, & Dey, 2011) adapted PSO to binary optimization problems in their work. The process was carried out by getting mod of the continuous values on base 2. Pampara et al. (Pampara, Engelbrecht, & Franken, 2006) reorganized Differential Evolution (DE) algorithm for binary optimization with angle modulation technique. Deng et al. (Deng, Zhao, Yang, Peng, & Wei, 2011) produced binary candidate solutions for DE using logical gates. Binary optimization problems are solved with their approach called BDE. Kashan et al. (Kashan, Nahavandi, & Kashan, 2012) proposed a new approach called DisABC by redesigning the Artificial Bee Colony (ABC) algorithm for binary optimization. DisABC uses the similarity measure between binary vectors to generate new candidate solutions. Mirjalili and Lewis (Mirjalili & Lewis, 2013) combined six different transfer functions with the binary PSO algorithm to compare the performance of transfer functions. Some of the transfer functions they propose are sigmoid-based and some are tangent-hyperbolic-based. Sigmoid-based

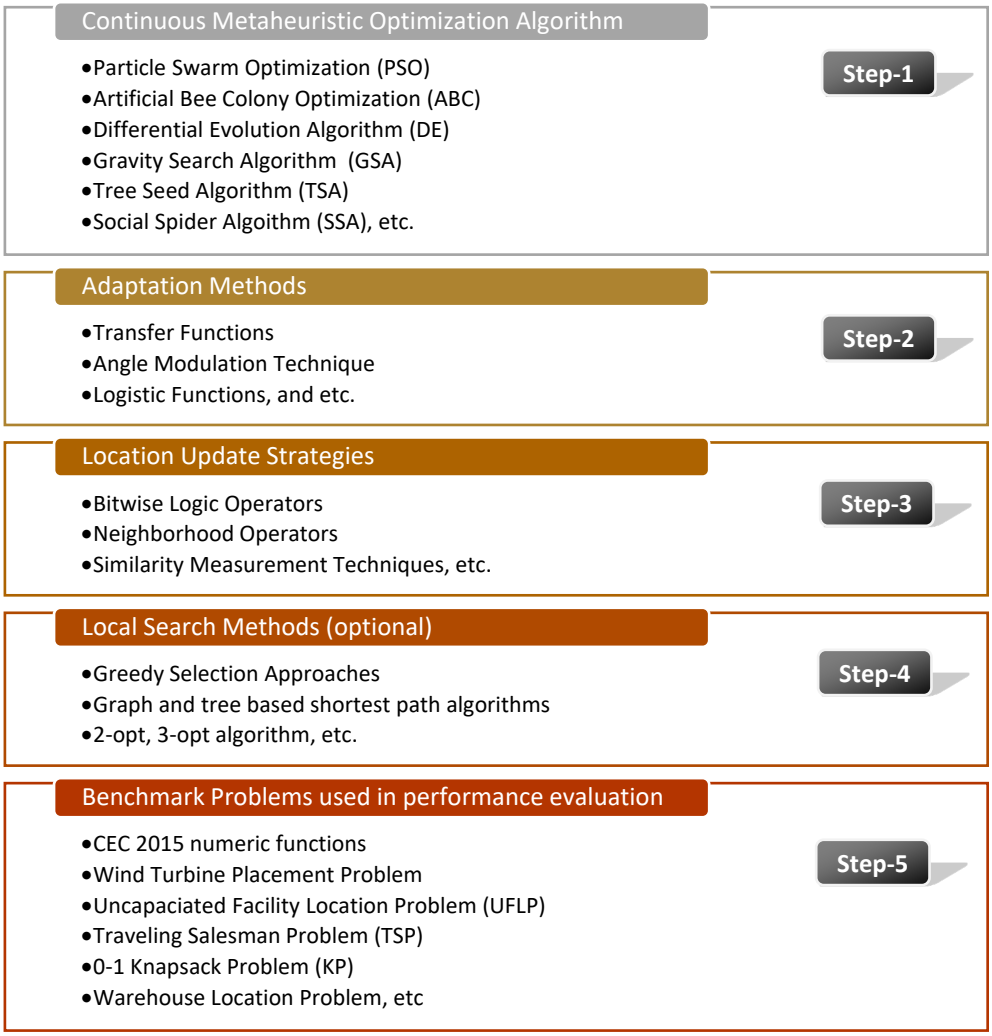
transfer functions are classified as S-Shaped, while others are classified as V-Shaped. Kiran (Kiran, 2015) binarized ABC with mode-based logistic functions and named ABCbin; here the bees working in continuous space were mapped by the mod process before being evaluated in the objective function. Cinar et al. (Cinar, Korkmaz, & Kiran, 2020) proposed the discrete Tree Seed Algorithm (TSA) by adding neighbourhood operators to the location update mechanism of the basic TSA to solve permutation-based optimization problems. Baş and Ülker (BaŞ & ÜLker, 2020) developed the Social Spider Algorithm (SSA)-based binary algorithm named BinSSA, for the solution of the feature selection problem, where transfer functions are used in the position update stage of the SSA algorithm. Basset et al. (Abdel-Basset, Mohamed, & Mirjalili, 2021) proposed a binary variant of the Equilibrium Optimizer algorithm using the S-shaped and V-shaped transfer functions. The researchers used the 0–1 knapsack problem to evaluate the performance of their proposed binary optimization algorithm. Pan et al. (Pan, Hu, & Chu, 2021) proposed a binary version of the fish migration optimization algorithm using transfer functions and applied it on unit commitment problems.

Considering the approaches proposed in the literature to convert the metaheuristic optimization algorithms, originally proposed for the solution of continuous optimization problems, into the solution of discrete optimization problems, we can summarize the adaptation process in 5 steps. The first step is to choose an algorithm that works in continuous search space, such as PSO, ABC, and DE, whose success has been recognized in continuous optimization problems. The second step is to integrate a matching method into the algorithm, which will ensure that the decision variables produced with continuous values due to the nature of the chosen algorithm are mapped to binary and/or integer values. The third step is to select or develop a location update strategy that can be applied on the candidate solution vectors, which are now expressible by permutation or binary coding. In the fourth step, an optional local search method is added to the algorithm to improve the performance of the algorithm and/or help increase the convergence speed. In the last step, the comparison test set to be used to analyse the success of the adapted algorithm is selected. Some of the frequently used test problems for the purpose are CEC 2015 numerical test functions, wind turbine placement problem (WTP), Uncapaciated Facility Placement Problems (UFLP), Traveling Salesman Problem (TSPLib) Sets, 0-1 Knapsack Problem, Warehouse Layout Problem, etc. The adaptation and development process can be expressed in five steps as shown in Figure 1 which is made within the scope of this study by the author to explain the subject clearly.

In this study, the focus is on step-2 and step-3 after the necessary research has been done. In these steps, certain adaptation methods such as transfer functions, angle modulation techniques, and logistic functions, which are frequently used in the literature, are discussed in detail. Then, popular location update strategies used in solving combinatorial optimization problems are explained in detail. The aim of the study is to present a tidy analysis of how continuous algorithms can be adapted for the solution of combinatorial optimization problems, which occupy an important place in the evolutionary optimization literature.

The remainder of this paper is organized as follows. Adaptation methods are briefly explained in section 2. The location update strategies are elaborated in section 3. Finally, concluding remarks and possible future works are provided in section 4.

Figure 1. Transforming Stages of Metaheuristic Algorithms for Discrete Optimization Problems



2. Adaptation Methods

For continuous optimization algorithms to solve binary problems, continuous values need to be converted to binary values (0 or 1) using some techniques. In this section, techniques commonly used by researchers in the literature for this purpose are mentioned. Transfer functions have an important place among these techniques. Also in this context, angle modulation technique, mode-based logistic functions, and similarity measurement techniques are explained.

2.1. Transfer Functions

The conversion of continuous Particle Swarm Optimization algorithm to binary algorithm by using the sigmoid function by Kennedy and Eberhart (Kennedy & Eberhart, 1997) in 1997 is one of the pioneering implementations of the use of transfer functions. Mirjalili and Lewis (Mirjalili & Lewis, 2013) contributed to this field by proposing 8 different transfer functions in 2013. These functions are named as S-shaped and V-shaped according to the shapes of the curve they form in the coordinate system. These functions are given in from Eq. (1) to Eq. (8).

$$\text{S1} \qquad T(x) = \frac{1}{1 + e^{-2x}} \tag{1}$$

$$\text{S2} \qquad T(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

$$\text{S3} \qquad T(x) = \frac{1}{1 + e^{\frac{-x}{2}}} \tag{3}$$

$$\text{S4} \qquad T(x) = \frac{1}{1 + e^{\frac{-x}{3}}} \tag{4}$$

$$\text{V1} \qquad T(x) = \left|\text{erf}\left(\frac{\sqrt{\pi}}{2}x\right)\right| \tag{5}$$

$$\text{V2} \qquad T(x) = |\tanh(x)| \tag{6}$$

$$\text{V3} \qquad T(x) = \left|\frac{x}{\sqrt{1 + x^2}}\right| \tag{7}$$

$$\text{V4} \qquad T(x) = \left|\frac{2}{\pi}\arctan\left(\frac{\pi}{2}x\right)\right| \tag{8}$$

The curves of S-shaped and V-shaped transfer functions are shown in Figure 2 and Figure 3, respectively.

Figure 2. S-shaped family of transfer functions (Mirjalili & Lewis, 2013)

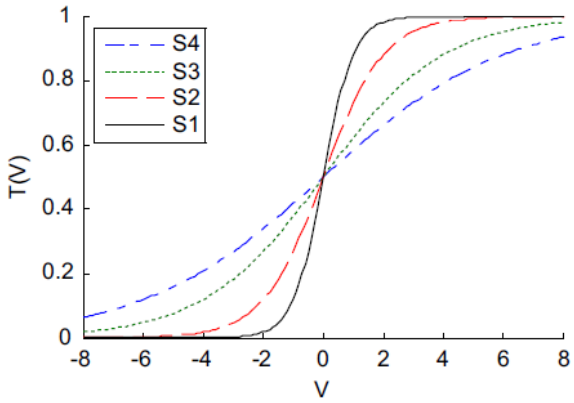
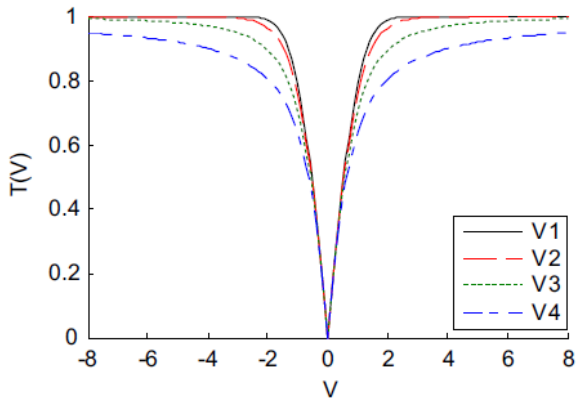


Figure 3. V-shaped family of transfer functions (Mirjalili & Lewis, 2013)



Apart from these 8 transfer functions, there are other transfer functions that have been proposed recently by modifying these functions for the same purpose. Such as Time-varying S-shaped Transfer functions, Linear Transfer function, U-Shaped Transfer function, Quadratic Transfer function (Beheshti, 2021).

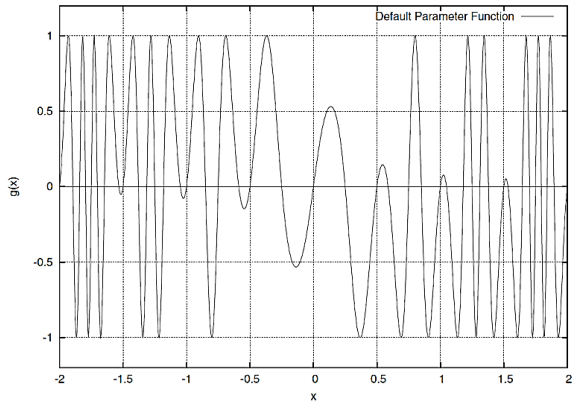
2.2. Angle Modulation

This method, which allows to map the values of decision variables from continuous space to binary space, is based on angle modulation used in the field of signal processing in the telecommunications industry. For the first time in metaheuristic algorithms, it was applied to PSO in order to develop a strategy for the Iterated Prisoners Dilemma (Franken & Engelbrecht, 2005). Its mathematical definition is a trigonometric function formed by the sin and cos functions. The formula of the technique is given in Eq. (9)

$$g(x) = \sin(2\pi(x - a) \times b \times \cos(A)) + d \tag{9}$$

where $A = 2\pi \times c(x - a)$ and x is a single element from a set of evenly separated intervals determined by the required number of bits that need to be generated. For example, a selection of 10 evenly spaced points between 0 and 10 in order to generate 10-bit values. The coefficients of Eq. (1) determine the shape of the generating function. The coefficient a represents the horizontal shift of the generating function, b represents the maximum frequency of the composed sin function, c represents the frequency of the composed cos function and d determines the vertical shift of the generating function. A generating function with the default coefficient values (a = 0, b = 1, c = 1, d = 0) is depicted in Figure 4.

Figure 4. Generating function with default values (a = 0, b =1, c = 1, d = 0)



A metaheuristic optimization algorithm is run on a 4-dimensional tuple representing the parameters (a, b, c, d) in this equation instead of optimizing the n-dimensional real bit string. After each iteration, these four parameters are replaced in Eq. (9). The resulting function is then sampled at equally spaced intervals to produce one bit for each interval with the set of all generated bits representing the binary vector solution of the original problem. Finally, to generate the real bit string, the sample points generator function is fed, and the output of Eq. (9) is calculated. If the resulting value is positive, the corresponding bit value is recorded as bit 1, otherwise the bit value is recorded as 0.

2.3. Mod Based Logistic Functions

To solve the binary optimization problem, the continuous values produced for the decision variables of the algorithm working in the continuous search space should be converted to binary values. The most basic approach developed for this purpose is to get the base-2 mode of the continuous values produced before being evaluated in the objective function. There are several studies using this approach in the literature (Kiran, 2015; Sevkli & Guner, 2006).

3. Location Update Strategies

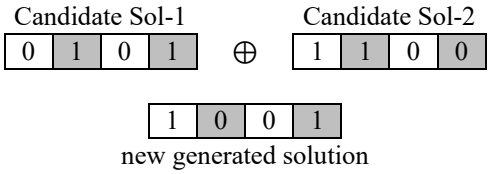
In this section, location update strategies that are frequently used in the literature in solving binary and/or combinatorial optimization problems are explained. These strategies are respectively: Bitwise operators, neighboring operators, and similarity measure techniques.

3.1. Bitwise Operators

In binary optimization, each dimension of a candidate solution only takes the value 1 or 0. The mutual dimensions of the two candidate solutions used for the location update process are inserted into the selected logic gate and the new dimension value is determined. Therefore, the logic gate used in algorithms is very important. As is known, there are four basic logic gates: AND, OR, NOT and XOR gates. If the OR gate is used for the location update operation, the corresponding size of the candidate solution will be 1 with 75% probability. If the AND gate is used for the update operation, the corresponding size of the candidate solution will be 1 with a 25% probability. If the XOR gate is used for the position update operation, the probability of choosing the values 0 and 1 is equal. For this reason, XOR gate is generally preferred in binary optimization algorithms developed based on gates.

Figure 5 shows a new candidate solution that will be obtained when two candidate solutions with 4-dimensions are subjected to a location update process using the XOR gate for all dimensions. Depending on the algorithm to be developed here, updating the dimensions can be carried out depending on a predefined probability ratio to keep the balance between exploration and exploration.

Figure 5. Location update process with XOR gate



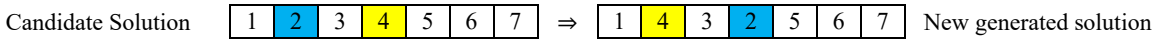
3.2. Neighbourhood Operators

It is one of the frequently preferred location update approaches for solving discrete integer problems. During the process of creating candidate solutions, there are 3 basic neighborhood operators: swap, shift and symmetry. In previous studies, these basic operators are used in different combinations in pairs and applied to optimization problems. Researchers have tried to develop the discrete optimization algorithm by determining the most suitable combination as a result of the runs (Ghosh, 2003; Zhou, Gao, Yang, & Gui, 2016).

3.2.1. Swap Operator

To generate a new candidate solution with the swap operator, two different random numbers are generated between [1, *D*] where D is the number of dimensions. The dimension values in these two positions in the candidate solution vector are swapped. The swap operator makes minor changes to the candidate solution vector. An example operation showing the operation of the swap operator is shown in Figure 6. In the example given, the number of dimensions is 7 and the dimensions to be swapped are specified as 2 and 4.

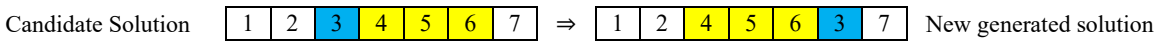
Figure 6. Implementation of Swap Operator



3.2.2. Shift Operator

Two different random numbers (x and y) are generated between [1, *D*] where D is the number of dimensions to generate a new candidate solution with the shift operator. Then the decision variable at position x is memorized, and the other decision variables are shifted to the left in the range [x, y]. Next, the memorized decision variable is assigned to the y location. The shift operator makes more changes to the candidate solution than the swap operator. An example of the implementation of the shift operator is shown in Figure 7. In this example, D equals to 7 and the dimensions to be shifted are determined as x=3 and y=6.

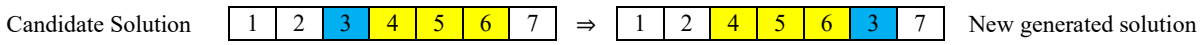
Figure 7. Implementation of Shift Operator



3.2.2. Symmetry Operator

Two different random numbers are generated between [1, *D*]. Two blocks with the same number of elements located in consecutive positions with respect to these points are randomly determined. Then, the two blocks are both swapped, and the blocks are inverted within themselves. The symmetry operator causes a large change in the candidate solution vector. An example operation showing the operation of the symmetry operator is shown in Figure 8. In the given example, D equals to 7 and the selected dimensions are determined as x=2 and y=6.

Figure 8. Implementation of Symmetry Operator



3.3. Similarity Measurement Techniques

This technique uses the difference measure of binary structures instead of the location update operators used in solving continuous optimization problems in metaheuristic algorithms and proposes a differential operator that preserves the main features of the algorithms. The main feature of this differential operator is that it works in continuous space while its results are used in discrete space. One of the main benefits of this approach is that it allows the structural knowledge of the problem to be used by using general or problem-related heuristic procedures during construction of the new solution (Kashan et al., 2012).

Measuring the similarity/dissimilarity of binary structures, which is frequently used for pattern representation, plays a critical role in many problems such as clustering and classification. The similarity measure shows how close or far the candidate solutions are from each other in stochastic optimization. Numerous similarity measurement techniques that can be applied for binary representation Choi et al. (Choi, Cha, & Tappert, 2010) was examined in detail in his study.

One of the most widely used general purpose similarity measures to determine the degree of similarity between feature vectors using binary representation is Jaccard's similarity coefficient (Choi et al., 2010; Hancer, Xue, Karaboga, & Zhang, 2015; Kashan et al., 2012). For this reason, the use of similarity measurement techniques in optimization algorithms will be explained through the Jaccard coefficient.

Let $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$ and $X_j = (x_{j1}, x_{j2}, \dots, x_{jD})$ be two candidate solution vectors whose similarity is to be measured where x_{id} ve x_{jd} $(\forall d = 1,2, \dots D)$ represent d. bit of X_i and X_j, respectively. The bits can be assigned to only values 0 and 1. The values that x_{id} and x_{jd} can take together are $S1 = (0,0),\ S2 = (0,1),\ S3 = (1,0),\ S4 = (1,1)$. The similarity measure is calculated by Eq. (10).

$$Similarity\ (X_i, X_j) = \frac{M_{11}}{M_{01} + M_{10} + M_{11}} \tag{10}$$

where M_{11} represents the total number of S1, M_{01} the total number of S2, M_{10} the total number of S3, and M_{00} the total number of S4. Therefore, it must be $M_{11} + M_{01} + M_{10} + M_{00} = D$. After calculation of the similarity, the dissimilarity calculation is made by Eq. (11).

$$\text{Dissimilarity}\ (X_i, X_j) = 1 - \ \text{Similarity}\ (X_i, X_j) = 1 - \frac{M_{11}}{M_{01} + M_{10} + M_{11}} \tag{11}$$

This dissimilarity value calculated for the solution of binary optimization problems is integrated into the location update strategies of metaheuristic algorithms designed for continuous search space. These strategies are mostly based on calculating the arithmetic difference between the parameters of the two candidate solutions. Calculation is applied by replacing the minus operator with the dissimilarity measure of binary vectors.

4. Conclusions

Optimization problems are categorized in two classes according to the values of the decision variables. These are continuous and discrete optimization. If the decision variables can take infinite values in a range defined by real numbers in the search space, the problem is defined as continuous optimization, while if the decision variables consist of discrete values to be assigned, the problem is defined as discrete optimization problem. Discrete problems are also generally grouped as binary and integer optimization problems. It is a binary problem if the decision variables can only take the values 0 and 1. On the other hand, if the decision variables can take one of n values after they are arranged to be expressed with integers, in other words, if a certain combination of these integers is desired, it is a combinatorial optimization problem. In this context, many real-world problems can be defined as discrete optimization problems. Although many deterministic and linear programming methods have been proposed in the literature to solve discrete optimization problems, it makes the solution difficult or impossible

for large-sized problems in exponential time. In this case, it is known that the use of metaheuristic algorithms is quite advantageous.

In this study, the approaches used for the use of metaheuristic optimization algorithms, which by their nature were developed only for the solution of continuous optimization problems, in the solution of discrete optimization problems are examined. Adaptation and location update strategies that are frequently used in the literature are explained in detail. These methods and approaches are presented to the reader categorically. During the development of new discrete algorithms, the steps of the adaptation process are clearly listed. These steps are (i) selection of the continuous optimization algorithm, (ii) determination of the adaptation method, (iii) development of the location update strategy, (iv) optionally integrating a local search method into the algorithm, and (v) performance analysis. In this way, a directional study has been put forward for researchers working in this field.

In further studies, a detailed examination of the local search strategies that can be used in discrete optimization algorithms, which is the 4th step of the process, and the benchmark problems, which is the 5th step, will be made and the effects of the algorithms on their performance will be examined.

References

Abdel-Basset, M., Mohamed, R., & Mirjalili, S. (2021). A Binary Equilibrium Optimization Algorithm for 0–1 Knapsack Problems. *Computers & Industrial Engineering, 151*, 106946. doi:https://doi.org/10.1016/j.cie.2020.106946

BaŞ, E., & ÜLker, E. (2020). An efficient binary social spider algorithm for feature selection problem. *Expert Systems with Applications, 146*, 113185. doi:https://doi.org/10.1016/j.eswa.2020.113185

Beheshti, Z. (2021). UTF: Upgrade transfer function for binary meta-heuristic algorithms. *Applied Soft Computing, 106*, 107346. doi:https://doi.org/10.1016/j.asoc.2021.107346

Choi, S.-S., Cha, S.-H., & Tappert, C. C. (2010). A survey of binary similarity and distance measures. *Journal of systemics, cybernetics and informatics, 8*(1), 43-48.

Cinar, A. C., Korkmaz, S., & Kiran, M. S. (2020). A discrete tree-seed algorithm for solving symmetric traveling salesman problem. *Engineering Science and Technology, an International Journal, 23*(4), 879-890. doi:https://doi.org/10.1016/j.jestch.2019.11.005

Deng, C., Zhao, B., Yang, Y., Peng, H., & Wei, Q. (2011). *Novel Binary Encoding Differential Evolution Algorithm*, Berlin, Heidelberg.

Franken, N., & Engelbrecht, A. P. (2005). Particle swarm optimization approaches to coevolve strategies for the iterated prisoner's dilemma. *IEEE Transactions on Evolutionary Computation, 9*(6), 562-579. doi:10.1109/TEVC.2005.856202

Ghosh, D. (2003). Neighborhood search heuristics for the uncapacitated facility location problem. *European Journal of Operational Research, 150*(1), 150-162. doi:https://doi.org/10.1016/S0377-2217(02)00504-0

Hancer, E., Xue, B., Karaboga, D., & Zhang, M. (2015). A binary ABC algorithm based on advanced similarity scheme for feature selection. *Applied Soft Computing, 36*, 334-348. doi:https://doi.org/10.1016/j.asoc.2015.07.023

Kashan, M. H., Nahavandi, N., & Kashan, A. H. (2012). DisABC: A new artificial bee colony algorithm for binary optimization. *Applied Soft Computing, 12*(1), 342-352. doi:https://doi.org/10.1016/j.asoc.2011.08.038

Kennedy, J., & Eberhart, R. C. (1997, 12-15 Oct. 1997). *A discrete binary version of the particle swarm algorithm.* Paper presented at the 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation.

Kiran, M. S. (2015). The continuous artificial bee colony algorithm for binary optimization. *Applied Soft Computing, 33*, 15-23. doi:https://doi.org/10.1016/j.asoc.2015.04.007

Mirjalili, S., & Lewis, A. (2013). S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization. *Swarm and Evolutionary Computation, 9*, 1-14. doi:https://doi.org/10.1016/j.swevo.2012.09.002

Pampara, G., Engelbrecht, A. P., & Franken, N. (2006, 16-21 July 2006). *Binary Differential Evolution.* Paper presented at the 2006 IEEE International Conference on Evolutionary Computation.

Pan, J.-S., Hu, P., & Chu, S.-C. (2021). Binary fish migration optimization for solving unit commitment. *Energy, 226*, 120329. doi:https://doi.org/10.1016/j.energy.2021.120329

Saha, S., Kole, A., & Dey, K. (2011). *A Modified Continuous Particle Swarm Optimization Algorithm for Uncapacitated Facility Location Problem*, Berlin, Heidelberg.

Sevkli, M., & Guner, A. R. (2006). *A Continuous Particle Swarm Optimization Algorithm for Uncapacitated Facility Location Problem*, Berlin, Heidelberg.

Yang, X.-S. (2010). *Engineering optimization: an introduction with metaheuristic applications*: John Wiley & Sons.

Zhou, X., Gao, D. Y., Yang, C., & Gui, W. (2016). Discrete state transition algorithm for unconstrained integer optimization problems. *Neurocomputing, 173*, 864-874. doi:https://doi.org/10.1016/j.neucom.2015.08.041