Hindawi
Complexity
Volume 2017, Article ID 8404231, 19 pages
https://doi.org/10.1155/2017/8404231

WILEY | Hindawi

Review Article

Putting Continuous Metaheuristics to Work in Binary Search Spaces

Broderick Crawford,[1] Ricardo Soto,[1] Gino Astorga,[1,2] José García,[1,3] Carlos Castro,[4] and Fernando Paredes[5]

[1]*Pontificia Universidad Católica de Valparaíso, 2362807 Valparaíso, Chile*
[2]*Universidad de Valparaíso, 2361864 Valparaíso, Chile*
[3]*Centro de Investigación y Desarrollo Telefónica, 7500961 Santiago, Chile*
[4]*Universidad Técnica Federico Santa María, 2390123 Valparaíso, Chile*
[5]*Escuela de Ingeniería Industrial, Universidad Diego Portales, 8370109 Santiago, Chile*

Correspondence should be addressed to José García; joseantonio.garcia@telefonica.com

Received 24 January 2017; Revised 30 March 2017; Accepted 9 April 2017; Published 11 May 2017

Academic Editor: Jia Hao

In the real world, there are a number of optimization problems whose search space is restricted to take binary values; however, there are many continuous metaheuristics with good results in continuous search spaces. These algorithms must be adapted to solve binary problems. This paper surveys articles focused on the binarization of metaheuristics designed for continuous optimization.

1. Introduction

In practically any activity that is performed, the resources are scarce; thus, we must properly utilize such resources. To this end, we can use technical optimization. Such problems are common in engineering, economics, machine learning, agriculture and, others areas. We found applications in learning automata in dynamic environments [1], optimum design of structures [2], load dispatch problems [3], optimization of directional overcurrent relay times [4], two-dimensional intermittent search processes [5], and control and risk monitoring [6], among other various real problems in industry.

Some models are logical only if the variables take on values from a discrete set, often a subset of integers, whereas other models contain variables that can take on any real value. Models with discrete variables are discrete optimization problems, and models with continuous variables are continuous optimization problems. In general, continuous optimization problems tend to be easier to solve than discrete optimization problems; the smoothness of the functions means that the objective function and constraint function values at a point x can be used to deduce information about points in a neighborhood of x. However, improvements in algorithms and in computing technology have dramatically increased the size and complexity of discrete optimization problems that can be solved efficiently.

Discrete optimization, that is, the identification of the best arrangement or selection of a finite number of discrete possibilities [7], has its origin in the economic challenge of efficiently utilizing scarce resources and effectively planning and managing operations. The decision problems in the field of operations management were among the first to be modeled as discrete optimization problems, for example, the sequencing of machines, the scheduling of production, or the design and layout of production facilities [8]. Today, discrete optimization problems are recognized in all areas of management when referring to the minimization of cost, time, or risk or the maximization of profit, quality, or efficiency [9]. Typical examples of such problems are variants of assignment and scheduling problems, location problems, facility layout problems, set partitioning and set covering problems, inventory control, and traveling salesman or vehicle routing problems [10], among others.

The difficulty level of such optimization problems is conceptualized by the theory of computational complexity [11, 12]. In this context, two complexity classes are of particular interest: P and NP (whereby the inclusion $P \subseteq$ NP holds). The problem class P contains all decision problems that can be solved in polynomial time in the size of the input on a deterministic sequential machine. These problems are considered to be easy and efficiently solvable [13]. The class NP contains all decision problems that can be solved in polynomial time on a nondeterministic machine.

A nondeterministic machine has two stages: the first one is the guessing stage and the second one is the checking stage. In the case of class NP, this checking stage is computable in polynomial time. A subclass of problems in NP is called NP-complete. A problem c is said to be NP-complete if it is a problem belonging to class NP and additionally has the feature that, given any other problem $n \in$ NP, n is polynomial time reducible to c. Finally a problem h of NP-hard type corresponds to a problem which is not necessarily NP but, given any problem $n \in$ NP, n is polynomial time reducible to h.

Many important discrete optimization problems are known to be NP-hard; that is, in the worst case, the time required to solve a problem instance to optimality increases exponentially with its size; hence, these problems are easy to describe and understand but are difficult to solve. Even for problems of moderate size, it is practically impossible to determine all possibilities to identify the optimum. Consequently, heuristic approaches, that is, approximate solution algorithms, are considered to be the only reasonable way to solve difficult discrete optimization problems. Accordingly, there is a vast and still growing body of research on metaheuristics for discrete optimization that aim at balancing the trade-off between computation time and solution quality [14].

Metaheuristics provide general frameworks for the creation of heuristic algorithms based on principles borrowed from classical heuristics, artificial intelligence, biological evolution, nervous systems, mathematical and physical sciences, and statistical mechanics. Although metaheuristics have proven their potential to identify high-quality solutions for many complex real-life discrete optimization problems from different domains, the effectiveness of any heuristic strongly depends on its specific design [15]. Hence, the abilities of researchers and practitioners to construct and parameterize heuristic algorithms strongly impact algorithmic performance in terms of solution quality and computation times. Consequently, there is a need for a deeper understanding of how heuristics need to be designed such that they achieve high effectiveness when searching the solution spaces of discrete optimization problems.

However, many of the well-known metaheuristics originally worked on continuous spaces because these can be formulated naturally in a real domain; examples of these metaheuristics are particle swarm optimization (PSO) [16], magnetic optimization algorithm (MOA) [17], cuckoo search (CS) [18], firefly algorithm (FA) [19], galaxy-based search (GS) [20], earthworm optimization (EW) [21], lightning search (LS) [22], moth-flame optimization (MF) [23], sine cosine (SC) [24], and black hole (BH) [25]. However, researchers have been developing binary versions that make these metaheuristics capable of performing in binary spaces. There are different methods for developing the binary version of a continuous heuristic algorithm while preserving the principles inspiring the search process. Examples of such binarizations are harmony search (HS) [26], the differential evolution algorithm (DE) [27–30], particle swarm optimization (PSO) [31], the magnetic optimization algorithm (MOA) [32], the gravitational search algorithm (GSA) [33], the firefly algorithm (FA) [34], the shuffled frog leaping algorithm (FLA) [35], the fruit fly optimization algorithm (FFA) [36], the cuckoo search algorithm (CSA) [37], the cat swarm optimization algorithm (CSOA) [38], the bat algorithm [39], the Black Hole Algorithm [40], the algae algorithm (AA) [41], and fireworks [42].

In contrast to the continuous binary approaches, we also found in the literature the inverse transformation, that is, from discrete techniques to continuous [43, 44]. This inverse approach uses the concepts of probability density function and its associated cumulative distribution function. When we use the inverse of cumulative distribution functions, we can produce uniformly distributed real numbers. This process is a general way to transform discrete metaheuristics to continuous metaheuristics. Typical probability density distributions were proposed in [45, 46].

This article is a review of the main binarization methods used when we are putting continuous metaheuristics to work in binary search spaces. The remainder of this paper is organized as follows. In Section 2, we present the main optimization problem definitions, the principal optimization techniques, and the different types of variables. Section 3 provides a definition of metaheuristics. In Section 4, we describe the main criteria for transforming continuous metaheuristics to discrete metaheuristics. Section 5 presents the most frequently used techniques allowing the binarization of the continuous metaheuristic. In the discussion in Section 6, we summarize and analyze the techniques and problems in terms of the number of articles published. The conclusions are outlined in Section 7, which presents a summary table that compares metaheuristics and discretization or binarization techniques.

2. Concepts and Notations

This section establishes the definitions and notations required for understanding the discretization and binarization techniques. For this purpose, we need to define some basic concepts.

2.1. Optimization Problem. The main goal of optimization metaheuristics is to resolve an optimization problem. An optimization problem corresponds to the pair of search space (S) and objective function (f). This pair is denoted as $O = (S, f)$, where S is generally a vector space, $S \neq \emptyset$, and $f : S \to \mathbb{R}$. Let $X = (x_1, \ldots, x_n) \in S$ be a feasible solution of the optimization problem. The solution of the optimization problem (S, f) when we are minimizing corresponds to finding a solution $X_j \in S$ such that $f(X_j) \leq f(X_i),\ \forall X_i \in S$.

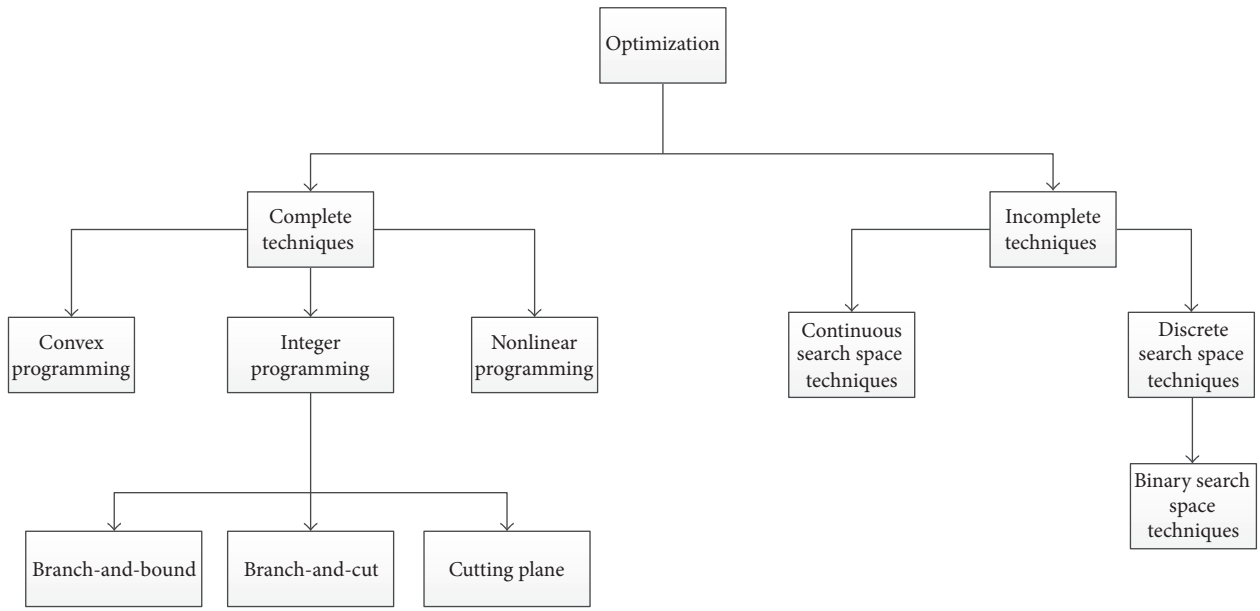

Figure 1: Optimization techniques.

In the case of a maximization problem, it can be transformed into a minimization problem by multiplying the objective function by −1.

2.1.1. Search Space. A search space, S, is a set of all possible points or solutions of the optimization problem that satisfy the problem's constraints. When we classify the parameters that make up each point of the solution, there are two groups. The first group corresponds to parameters with an unordered domain. These parameters do not have an exploitable structure; that is, they do not naturally have a metric, an order, or a partial order and therefore it is not feasible to use optimization methods to find optimal values. The only option for these cases is to use sampling [47]. A second group of parameters corresponds to those that naturally have a structure such as metric, order, or partial order. In this case we can take advantage of this structure to use optimization methods to find optimal values. Within this second group we often find parameters of real, discrete, or binary type. In terms of these real, discrete, or binary parameters the optimization problems can be classified as real optimization problems ($S \subset \mathbb{R}^n$), discrete optimization problems ($S \subset \mathbb{Z}^n$), binary problems ($S \subset \mathbb{B}^n$), and mixed problems. Since our review is directly linked to continuous, discrete, and binary optimization methods, from now on we will focus on these types of parameters.

2.1.2. Neighborhood. Let (S, f) be an optimization problem. A neighborhood structure is a function:

$$N : S \longrightarrow P(S), \tag{1}$$

where $P(S)$ is a power set of S. N function assigns a set $N(X_i) \in P(S)$ for each $X_i \in S$ element, where $N(X_i)$ is the neighborhood of X_i.

2.1.3. Local Optimum. Let (S, f) be an optimization problem and $S_j \subset S$ be the neighborhood of $X_j \in S$, $S_j = N(X_j)$. X_j is a local optimum (minimum) if it satisfies the following inequality:

$$f\left(X_j\right) \leq f\left(X_i\right), \quad \forall X_i \in S_j. \tag{2}$$

2.2. Optimization Techniques. There are several optimization techniques, as shown in the overview in Figure 1. We can group them into complete techniques and approximate or incomplete techniques. Without pretending to be exhaustive in the classification, we mentioned to our understanding the main techniques, giving more detail to the case of complete techniques in integer programming due to the proximity with the combinatorial problems. The exact or complete techniques are those where we find an optimum result regardless of the process time. For integer programming, the typical techniques are branch-and-cut and branch-and-bound. Many combinatorial optimization problems can be formulated as mixed integer linear programming problems. They can then be solved using branch-and-cut or branch-and-bound methods, which are exact algorithms that consist of a combination of a cutting plane method with a branch-and-bound algorithm. These methods work by solving a sequence of linear programming relaxations of the integer programming problem. Cutting plane methods improve the relaxation of the problem to more closely approximate the integer programming problem, and branch-and-bound algorithms proceed by a sophisticated divide-and-conquer approach to solve problems. Unfortunately when the problems are NP-hard and the size of instance grows, these algorithms do not provide good results. On the other hand, the incomplete techniques are those where a good solution is found that is not necessarily the best but found in a short processing time. This technique better fits the

actual conditions of the problems since, in daily life, the solutions of the problems are required in a given time. Within the approximate or incomplete techniques, we find metaheuristics.

In general terms, a metaheuristic attempts to find values for the variables that will provide an optimum value of the objective function.

2.3. Search Space. As our focus is on the continuous, discrete, and binary searches spaces (see Section 2.1, search space), a solution in that context can be classified into three categories, as follows [48]:

(i) *Continuous Variable*. Continuous variables are when the variable can have any value in the given interval.

(ii) *Discrete Variable*. Discrete variables correspond to variables that may have integer or binary values.

(iii) *Mixed Variables*. In this case, the variables can have many real, integer, or binary values; thus, it is called a mixed problem.

Discrete variables arise in many optimization problems, for example, in manufacturing [49], cutting and packing problems [50], integer programming [51], and quadratic assignment [52]. A common reason for discrete variables occurring is when the resources of interest are quantified in terms of integers or Boolean values, for example, in production lines, scheduling processes, or resource assignments. There is a set of classic problems that can be treated in binary form, such as the well-known knapsack problem [53], the set covering problem [54], and the traveling salesman problem [55].

For example, in the knapsack problem, the jth item has weight w_j and value c_j. The objective is to maximize the total value of the items placed in the knapsack subject to the constraint that the weight of the items does not exceed a limit b. To formulate this problem, one can let x_j be the binary variable such that

$$x_j = \begin{cases} 1 & \text{if item } j \text{ is placed in the knapsack,} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Then, we want to maximize $c^T x$ subject to $w^T x \leq b$, where $x \in \{0, 1\}$.

Another situation where the use of discrete variables is appropriate is when we need to manage constraints that involve logical conditions. For example, suppose that we want $x_1 \geq 0 \Leftrightarrow x_2 \leq 0$ and $x_2 \geq 0 \Leftrightarrow x_1 \leq 0$ and that we also want to preserve the linearity of the problem. This can be achieved by including the linear constraints

$$\begin{aligned} -M(1-y) \leq x_1 \leq My \\ -My \leq x_2 \leq M(1-y), \end{aligned} \tag{4}$$

where y is a binary variable and M is a sufficiently large positive number that does not affect the feasibility of the problem. By this definition of M, if $y = 1$, then we will have $x_1 \geq 0$ and $x_2 \leq 0$, whereas if $y = 0$, we will have $x_2 \geq 0$ and $x_1 \leq 0$.

Another common situation that requires integer variables is when the problem involves set-up costs. As an example, consider a generator that supplies electricity to a local region with I nodes for T periods. Suppose that at period t the generator incurs a cost of s_t when it is turned on, a cost of p_t for producing electricity after it is turned on, a cost of s_i for supplying electricity to node i after it is turned on, and a cost of d_t for shutting it down. For $t \in \{1, 2, \ldots, T\}$, x_t, y_t, and z_t denote the binary variables such that

$$\begin{aligned} x_t &= \begin{cases} 1 & f \text{ generator is turned on in period } t, \\ 0 & \text{otherwise} \end{cases} \\ y_t &= \begin{cases} 1 & f \text{ generator is operating in period } t, \\ 0 & \text{otherwise} \end{cases} \\ z_t &= \begin{cases} 1 & f \text{ generator is shut down in period } t, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{5}$$

If we let w_{it} be variables that represent the percentage of the generator's capacity c_i for node $i = \{1, 2, \ldots, I\}$ that is used in period t, then the total costs incurred would be $\sum_{t=1}^{T}(s_t x_t + p_t y_t + d_t z_t + \sum_{i=1}^{I} c_i s_i w_{it})$. The objective is to minimize the total costs.

3. Metaheuristics

A metaheuristic is formally defined as an iterative generation process that guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, and learning strategies are used to structure information to efficiently find near-optimal solutions [14, 56]. The fundamental properties that characterize the set of metaheuristic algorithms are as follows:

(i) Metaheuristics are higher level strategies that guide the search process.

(ii) The goal is to efficiently explore the search space to find (quasi)optimal solutions.

(iii) Metaheuristic algorithms are approximate and generally nondeterministic.

(iv) The basic concepts of metaheuristics permit an abstract level of description.

(v) Metaheuristics are not problem specific.

(vi) Metaheuristics may utilize domain-specific knowledge in the form of heuristics that are controlled by the upper level strategy.

(vii) Today, more advanced metaheuristics use search experience (embodied in some form of memory) to guide the search.

Table 1: Example of random-key encoding scheme. I represents the random-key, and Z is the decodification solution.

I	0.15	0.56	0.99	0.12	0.45	0.76	0.73	0.87	0.95
Z	2	4	9	1	3	6	5	7	8

3.1. Metaheuristic Classification

(i) *Nature Inspired versus Non-Nature Inspired Algorithms*. Generally, it is the most natural way to classify metaheuristics since it is based on the origins of the algorithm. It takes into account whether their models have been inspired by nature. There are bioinspired algorithms, such as genetic algorithms (GAs) and ant colony optimization (ACOs), and non-nature inspired ones, such as tabu search (TS) and iterated local search (ILS). This classification is not very meaningful following the emergence of hybrid algorithms.

(ii) *Population-Based versus Single-Point Search (Trajectory)*. In this case, the characteristic used for the classification is the number of solutions used at the same time. On the one hand, single-point search algorithms work on a single solution describing a trajectory in the search space during the search process. They encompass local search-based metaheuristics, such as variable neighborhood search (VNS), tabu search (TS), and iterated local search (ILS). On the other hand, population-based methods work on a set of solutions (points) called a population.

(iii) *Static versus Dynamic Objective Function*. The algorithms that keep the objective function given in the problem during the entire process are called metaheuristics with static objective functions. However, there are other algorithms with dynamic objective functions, such as guided local search (GLS), which modify the fitness function during the search, incorporating information collected during the search process to escape from local optima.

Many of the metaheuristic techniques are motivated in an $\mathbb{R}^n$ vectorial space [16, 57–61]; naturally, they cannot solve discrete or binary optimization problems. Many methods have been proposed that allow the use of a real optimization metaheuristic in discrete or binary problems. These methods are called discretization if the method allows adapting the real technique to solve integer problems and called binarization if the method solves binary optimization problems. In the next sections, we propose and explain a grouping of the main discretization and binarization techniques.

4. Discretization of Continuous Metaheuristics

There are many problems that require discrete search spaces [62–64]. While investigating these techniques, we found many names. However, these techniques can be classified into three main groups:

(i) Rounding off generic technique.

(ii) Priority position techniques associated with scheduling problems.

(iii) Specific techniques associated with metaheuristic discretizations.

4.1. Rounding off Generic Techniques. This approach is one of the most commonly used approaches for managing discrete variables due to its simplicity and low computational cost. It is based on the strategy of rounding to the nearest integer. It was first used in [65] in the optimization of reactive power and voltage control as a discretization method.

The rounding off operator transforms $\mathbb{R}^n$ feasible solution into $\mathbb{Z}^n$ feasible solution. The metaheuristic operators are used without modifications, and two strategies exist to implement the discretization. The first strategy applies a rounding off near integer operation to the feasible solution in every iteration. In the second approach, it is applied at the end of the optimization process.

There are multiple problems that use this method, for example, optimization of transport aircraft wing [64], task assignment problem [62], and distribution systems reconfiguration [63]. The main metaheuristics that use the round-off method are ant colony [63], PSO [62, 64, 65], firefly [66], and artificial bee colony [66, 67]. The disadvantages of this method include the possibility that the solution is in a nonfeasible region. Moreover, the value of the fitness in the rounded point can be very different from that in the original point.

4.2. Priority Position Techniques: Random-Key or Small Value Position. The random-key encoding scheme is used to transform a position in a continuous space into a position in a discrete space. The random-key was first used in a scheduling problem in [68], where the solutions are elements of a $\mathbb{Z}^n$ space.

Let us start with a solution $X \in S$ of n dimensions. In each position, a random number in $[0, 1]$ is assigned, obtaining an I real random-key solution. To decode the position from I real random-key solution in a discrete space, the positions are visited in ascending order, generating a $Z \in \mathbb{Z}^n$ discrete solution. An example is shown in Table 1.

This method has been used with the gravitational search algorithm [68, 69], resolving the traveling salesman problem and scheduling problem. In both cases, the result of the gravitational algorithm is mapped as a random-key, where small values map to the top position.

The same method, but called small value position (SVP), was used for the first time in [70] to solve the single-machine total weighted tardiness problem using a PSO algorithm. Later, [71] utilized SVP with the firefly algorithm to schedule

jobs on grid computing, and [72] used SVP in the cuckoo search algorithm to resolve the design optimization of reliable embedded systems. Additionally, we found this method in the first steps of great value priority. This binarization technique is explained in Section 5.

4.3. Metaheuristic Discretization. This method has been used in [73] to solve distribution system reconfiguration in the presence of distributed generation using a teaching-learning metaheuristic.

Let us start with a continuous metaheuristic that produces values in $[0, 1]$ or we can adapt a metaheuristic using functions that map $\mathbb{R}$ to $[0, 1]$, for example, transfer function (see Section 5).

Let N be the number of elements of our problem and N correspond to the dimension of the search space. C is a result of the metaheuristic; then $C \in [0, 1]$, and multiply C by N, obtaining $\alpha = 1 + N \times C$ and $\alpha \in [1, N + 1]$. This real number is discretized by applying the following equation:

$$\beta = \min\left(\lfloor \alpha \rfloor, N\right). \tag{6}$$

This procedure allows us to map to discrete values between 1 and N. This has been applied to smart grids using teaching-learning-based optimization [73].

5. Binarization of Continuous Metaheuristics

In our study and the conceptualization of binarization techniques, we found two main groups of binarization techniques. The first group of techniques we call two-step binarization; these techniques allow working with the continuous metaheuristics without operator modifications and include two steps after the original continuous iteration; these two steps make the binarization of the continuous solution. The second group of techniques is called continuous-binary operator transformation; it redefines the algebra of the search space, thereby reformulating the operators.

5.1. Two-Step Binarization Technique. This technique works with the continuous operators without modifications. To perform the binarization, two additional steps are applied. The first step corresponds to introducing operators that transform the solution from $\mathbb{R}^n$ to $\{InterSpace\}$. For example, in great value priority, our interspace is $\mathbb{Z}^n$; in the case of a transfer function (TF), we have $[0, 1]^n$ and {Space functions} in angle modulation. The second step transforms from the interspace ($\mathbb{Z}^n$, $[0, 1]^n$, {Space functions}) into a binary space. A general scheme is presented in Figure 2.

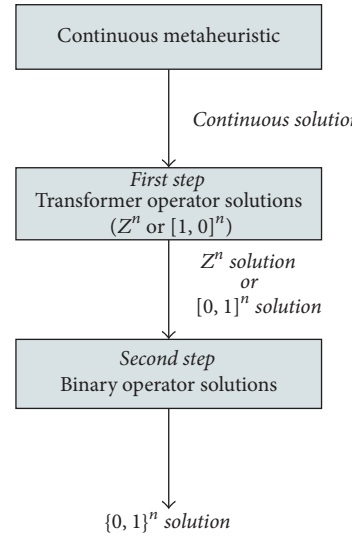

Figure 2: A general scheme for two-step binarization methods.

5.1.1. Transfer Function and Binarization

Transfer Function. In this technique, the first step corresponds to the transfer function, which is the most used normalization method and was introduced in [31]. The transfer function is a very cheap operator, and its range provides probability values and attempts to model the transition of particle positions. This function is responsible for the first step of the binarization method and mapping $\mathbb{R}^n$ solutions in $[0, 1]^n$ solutions.

In our revision, we found that two types of functions have been used in research: the S-shape [74–77] shown in (7) to (10), and their shapes are shown in Figure 3(a), and V-shape [54, 78, 79] shown in (11) to (14), and their shapes are shown in Figure 3(b).

$$T\left(d_w^j\right) = \frac{1}{1 + e^{-2d_w^j}} \tag{7}$$

$$T\left(d_w^j\right) = \frac{1}{1 + e^{-d_w^j}} \tag{8}$$

$$T\left(d_w^j\right) = \frac{1}{1 + e^{-d_w^j/2}} \tag{9}$$

$$T\left(d_w^j\right) = \frac{1}{1 + e^{-d_w^j/3}} \tag{10}$$

$$T\left(d_w^j\right) = \left|\operatorname{erf}\left(\frac{\sqrt{\pi}}{2} d_w^j\right)\right| = \left|\frac{\sqrt{2}}{\pi} \int_0^{(\sqrt{\pi}/2)d_w^j} e^{-t^2} dt\right| \tag{11}$$

$$T\left(d_w^j\right) = \left|\tanh\left(d_w^j\right)\right| \tag{12}$$

$$T\left(d_w^j\right) = \left|\frac{d_w^j}{\sqrt{1 + {d_w^j}^2}}\right| \tag{13}$$

$$T\left(d_w^j\right) = \left|\frac{2}{\pi} \arctan\left(\frac{\pi}{2} d_w^j\right)\right|. \tag{14}$$

Let $X = (x_1, \ldots, x_n)$ be a feasible solution to the problem; for each dimension, it is applied to the transfer function (TF), $i_j = \mathrm{TF}(x_j)$, obtaining an intermediate solution $I = (i_1, i_2, \ldots, i_n)$, where $I \in [0, 1]^n$. This transfer function defines the probability of changing a position (an assignment of a value to a variable or enumeration). According to [80], some concepts should be taken into account when selecting

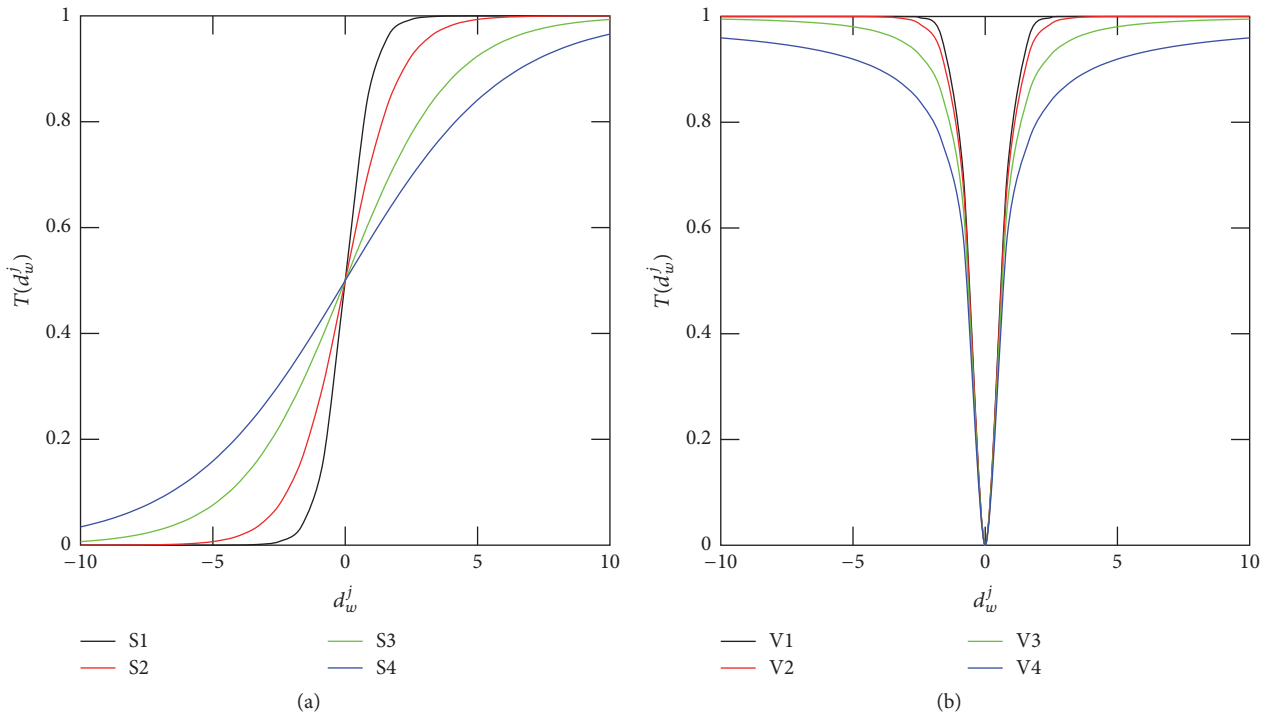

FIGURE 3: (a) S-shape and (b) V-shape and transfer functions.

a transfer function to map velocity values to probability. Intuitively, an S-shaped transfer function should provide a high probability of changing the position for a large absolute value of the velocity. Particles that have large absolute values for their velocities are probably far from the best solution; thus, they should switch their positions in the next iteration. It should also present a small probability of changing the position for a small absolute value of the velocity.

Binarization. The second step is the binarization technique, in which the particle I is transformed into a binary solution $B = (b_1, \ldots, b_n)$ by applying a binarization rule. In the literature, we found the binarization equations (15) to (18). In the following, rand is a random number in $[0, 1]$.

Standard. If the condition is satisfied, the second step operator returns the value 1, independent of the previous value. Otherwise, it returns 0.

$$x_{\text{new}}^j = \begin{cases} 1 & \text{if rand} \leq T\left(d_w^j\right) \\ 0 & \text{else.} \end{cases} \tag{15}$$

Complement. If the condition is satisfied, the second step operator returns the complement of the actual value.

$$x_{\text{new}}^j = \begin{cases} \text{complemento}\left(x_w^j\right) & \text{if rand} \leq T\left(d_w^j\right) \\ 0 & \text{else.} \end{cases} \tag{16}$$

Static Probability. A static α probability transition value is generated, and it is evaluated with a transfer function.

$$x_{\text{new}}^j = \begin{cases} 0 & \text{if } T\left(d_w^j\right) \leq \alpha \\ x_w^j & \text{if } \alpha < T\left(d_w^j\right) \leq \dfrac{1}{2}(1+\alpha) \\ 1 & \text{if } T\left(d_w^j\right) \geq \dfrac{1}{2}(1+\alpha). \end{cases} \tag{17}$$

Elitist. This discretization method selects the value of the best individual of the population.

$$x_{\text{new}}^j = \begin{cases} x_b^j & \text{if rand} < T\left(d_w^j\right) \\ 0 & \text{else.} \end{cases} \tag{18}$$

Elitist Roulette. This discretization method, also known as Monte Carlo, selects the new value randomly among the best individuals of the population, as shown in Figure 4, with a probability proportional to its fitness.

In particle swarm optimization, this approach was first used in [31]; in [81], it was used to optimize the sizing of capacitor banks in radial distribution feeders; in [82], it was used for the analysis of bulk power system; and, in [83], it was used for network reconfiguration. This approach was also used by Crawford et al. to resolve the set covering problem using the binary firefly algorithm [54] and artificial bee colony algorithm [84] and in [84] to resolve the set covering problem with the artificial bee colony algorithm,

Table 2: The vector X corresponds to a continuous solution, the vector I is the great value position, and B is the result of the binarization rule.

X	6.2	7.3	2.4	7.8	9.1	2.5	6.9	5
I	5	3	8	2	1	7	4	6
B	1	0	1	1	0	1	0	1

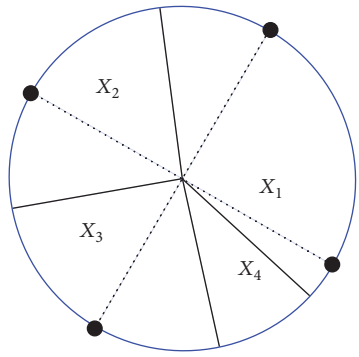

Figure 4: Elitist roulette.

and it was also used in [37] using the cuckoo search algorithm applied to the set covering problem. To resolve the unit commitment problem, [77] used firefly and PSO. The knapsack cryptosystem was approached in [74], the network and reliability constrained problem was solved in [78], and knapsack problems were solved in [41], all using the firefly algorithm. In [85], a teaching-learning optimization algorithm was used for designing plasmonic nanobipyramids based on the absorption coefficient.

5.1.2. Great Value Priority and Mapping

Great Value Priority. Great value priority (GVP) was introduced in [86] to solve a quadratic assignment problem by applying a particle swarm optimization (PSO) algorithm. This method encodes a continuous space $\mathbb{R}^n$ into the binary space $\{0, 1\}^n$ having two principal properties: it is an injective mapping, and it reflects a priority order relation, which is suitable for assignment problems.

Let us start with the solution $X = (x_1, x_2, \ldots, x_n)$; then, as a first step, we obtain a permutation I that lies in $\mathbb{Z}^n$. The GVP rule chooses the heaviest element and places its position in the first element of I. For the remaining elements, choose the heaviest and place it in the second position and so on until the elements of the original vector are browsed.

Binarization. In this technique, the second step maps I to B. To obtain B, we apply the mapping rule shown in (19). The result is a binary solution B of n dimensions. A concrete example is presented in Table 2.

$$x_j = \begin{cases} 1 & \text{if } p_j > p_{j+1} \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

This technique has been used in other types of binary problems; for example, in [87], it was used to solve the antenna positioning (AP) problem using a binary bat algorithm. In this solution, the algorithm preserves the original operators and Euclidean geometry of the space and adds a new module that maps real-valued solutions into the binary ones. Unlike the quadratic binary algorithm, where the priority is intrinsic to the problems, in the AP problem, the use of priority is clearly not suitable. The result was not conclusive; there were good and bad solutions for different instances.

5.1.3. Angle Modulation and Rule

Angle Modulation (AM). This approach was used in the telecommunications industry for phase and frequency modulation of the signal [88]. This method uses a trigonometric function that has four parameters, and these parameters control the frequency and shift of the trigonometric function.

$$g_i\left(x_j\right) = \sin\left(2\pi\left(x_j - a_i\right) b_i \cos\left(2\pi\left(x_j - a_i\right) c_i\right)\right) + d_i. \tag{20}$$

In binary heuristic optimization, this method was first applied in PSO using a set of benchmark functions [89].

Consider an n-dimensional binary problem, and let $X = (x_1, x_2, \ldots, x_n)$ be a solution. We start with a four-dimensional search space, where each dimension represents a coefficient of (20).

In the first step from the four-dimensional space, we obtain a function in a function space. Specifically, from every solution (a_i, b_i, c_i, d_i) in this space, we obtain a g_i trigonometric function that lies in a function space.

Binarization. In the second step, for each single element x_j, apply rule 24 and obtain an n-dimensional binary solution.

$$b_{ij} = \begin{cases} 1 & \text{if } g_i\left(x_j\right) \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

Then, for each initial 4-dimensional solution (a_i, b_i, c_i, d_i), we obtain a binary n-dimensional solution $(b_{i1}, b_{i2}, \ldots, b_{in})$ that is a feasible solution of our n-binary problem.

The angle modulation technique was applied to network reconfiguration problems in [90] using a binary PSO method. This technique was also applied in [91] to a multiuser detection technique using a binary adaptive evolution algorithm. The antenna position problem was solved in [87] using an angle modulation binary bat algorithm. Using PSO in [92], they solved the N-queens problems, and, in [90], Liu et al. solved large-scale power systems. In [93], a differential evolution algorithm was applied to knapsack problems, and,

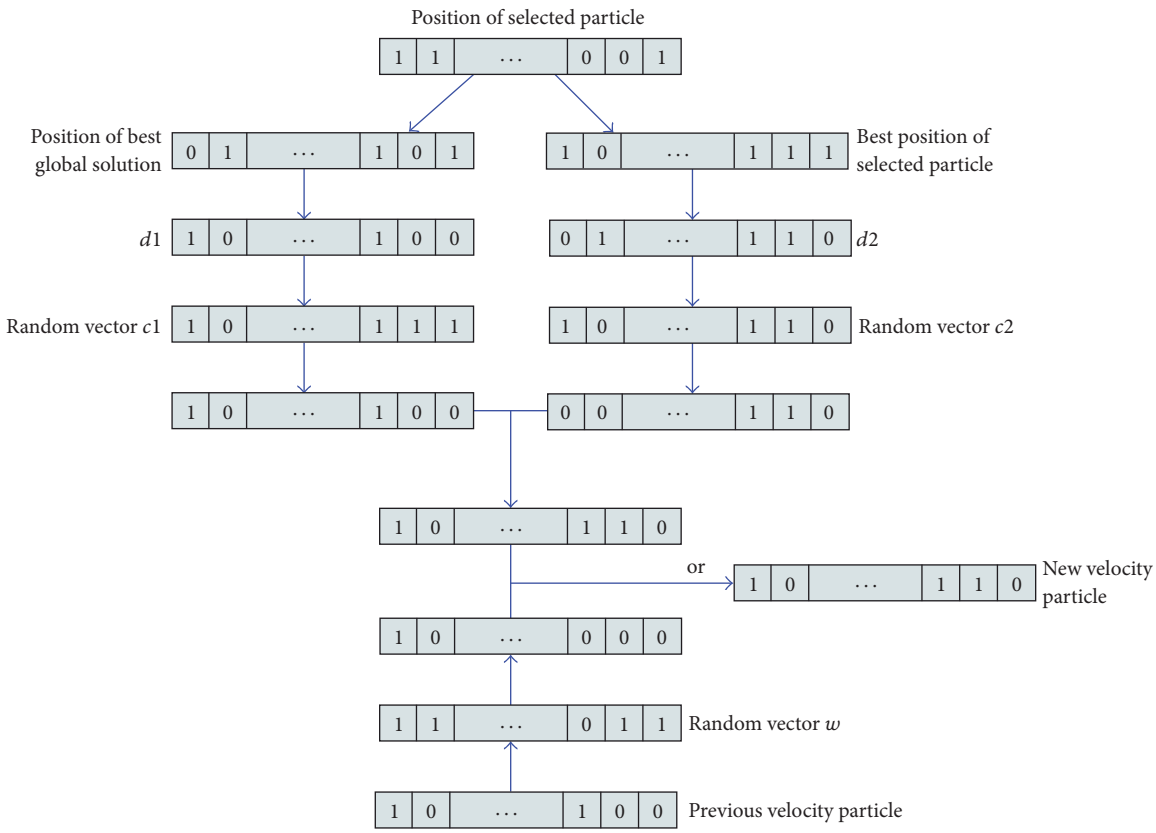

Figure 5: Example of Boolean approach.

in [94], it was applied to binary problems. Artificial bee colony was applied to a feature selection in [95] and to binary problems in [96].

5.2. Continuous-Binary Operator Transformation. These methods are characterized to redefine the operators of the metaheuristic, and there are two main groups. We call the first group modified algebraic operations. In this group, the algebraic operations of the search space are modified, and examples include the Boolean approach and set approach. The second group is called promising regions, and the operators are restructured in terms of selected promising regions in the search spaces. This selection is performed using a probability vector. Examples of this group include the quantum-based binary approach and binary method based on the estimation of distribution.

5.2.1. Modified Algebraic Operations

Boolean Approach (BA). This method belongs to modified algebraic operations. Let us transform the real operators into binary operators. This transformation is performed using Boolean operations. The operators act over the binary solutions. This approach emerged as a binarization technique of particle swarm optimization [97]; subsequently, [98] incorporated the inertia weight.

In Figure 5, we show a concrete example of applying the velocity Boolean equation (22) to the position of selected particle X_i. The Boolean notation is as follows: "XOR" = $\otimes$, "AND" = $\oplus$, and "OR" = $\odot$. The velocity and position Boolean Equations with inertia weight are presented in (22) and (23), respectively. $P_{\text{best},i}$ and P_{global} belong to the best position of the selected particle and the position of the global best solution, respectively, and c_1 and c_2 are random vectors. $V_i(t)$ corresponds to the velocity at time t.

$$V_i(t+1) = w \oplus V_i(t) \odot c_1 \otimes \left(P_{\text{best},i} \oplus X_i\right) \odot c_2 \otimes \left(P_{\text{global}} \oplus X_i\right) \tag{22}$$

$$X_i(t+1) = X_i(t) \oplus V_i(t+1). \tag{23}$$

This method has been applied to different binary optimization problems using the particle swarm method [97–99]. In [100], it was used with bitwise operations applied to optimization problems. The Boolean approach introduced an efficient velocity bounding strategy based on negative thymic selection of T-cells.

Set-Based Approach (SBA). The set-based approach is a technique that belongs to modified algebraic operations. It is a good framework for discrete problems. In this approach, we eliminate all structures of the space (vectorial or metrics),

and we work with a pure set. In a set, we have the standard set operations, union, intersection, complement, and so forth. Then, we need to redefine the operations of sums, multiplications, and others of the vector spaces and use set operations. Consequently, we have to reformulate the operators as discrete operators.

In the literature, there are numerous set frameworks applied to PSO algorithms. Reference [101] proposed a generic set-based algorithm, which had the same problems with the size of the positions and velocities. In [102], a generic set PSO algorithm was proposed, but its performance is less than that of other algorithms. In [103], an algorithm called S-PSO was proposed that can be used to adjust a continuous PSO to a discrete one. In [104], an SBPSO was proposed.

In a general framework, it is necessary to define some operations; let $\mathbb{P}(U)$ be a power set of U, and $(+,-)$ indicates whether the elements are added or removed in any operation. The definition of velocity is a transformation that maps a position to new positions.

(1) The addition of two velocities: $\oplus : \mathbb{P}((+,-) \times U)^2 \rightarrow \mathbb{P}((+,-) \times U)$, where $V_1 \oplus V_2 = V_1 \cup V_2$.

(2) The difference between two positions: $X_1 \ominus X_2$ is a mapping $\ominus : \mathbb{P}(U)^2 \rightarrow \mathbb{P}((+,-) \times U)$, where

$$X_1 \ominus X_2 = \left((+) \times \left(X_1 \setminus X_2\right)\right) \cup \left((-) \times \left(X_2 \setminus X_1\right)\right). \quad (24)$$

(3) Multiplication of a velocity by a scalar: $\otimes : [0,1] \times \mathbb{P}((+,-) \times U) \rightarrow \mathbb{P}((+,-) \times U)$, where the mapping is defined by picking a subset of $\lfloor \mu \times |V| \rfloor$ elements at random from velocity V.

(4) Addition of velocity and position: $\odot : \mathbb{P} \times \mathbb{P}((+,-) \times U) \rightarrow \mathbb{P}(U)$, where $X \ominus V = V(X)$.

Using these operations, our equations must be modified:

$$V_i(t+1) = w \otimes V_i(t) \oplus c_1 \otimes \left(P_{\text{best},i} \ominus X_i\right) \oplus c_2 \otimes \left(P_{\text{global}} \ominus X_i\right) \quad (25)$$

$$X_i(t+1) = X_i(t) \odot V_i(t+1).$$

This method modifies the operators velocity and position, and the construction is not simple. There are many variations of (25) [101, 103, 105], and in most cases they apply to PSO. In PSO, this technique has been used to solve the traveling salesman problem [106], the multidimensional knapsack problem, and vehicle routing problems [101]. The Boolean approach is a particular case of the set approach.

5.2.2. Promising Regions

Quantum Binary Approach. This approach, which belongs to promising regions, has been developed in PSO [107, 108]. It has been inspired in the uncertainly principle, where we cannot simultaneously determine position and velocity. Therefore, for individual particles, the PSO algorithm works in a different fashion, and we need to rewrite the operators.

In the quantum approach, each feasible solution has a position $X = (x_1, x_2, \ldots, x_n)$ and a quantum vector $Q = [Q_1, Q_2, \ldots, Q_n]$. Q_j represents the probability of x_j taking the value 1. For each dimension j, a random number between $[0,1]$ is generated and compared with Q_j; if rand $< Q_j$, then $x_j = 1$; otherwise, $x_j = 0$.

Then, the new P_{best} and P_{global} are calculated using the objective function. Finally, we update the transition probability using

$$Q_{\text{self}}(t) = \alpha P_{\text{best}}(t) + \beta\left(1 - P_{\text{best}}\right)$$

$$Q_{\text{social}}(t) = \alpha P_{\text{global}}(t) + \beta\left(1 - P_{\text{global}}\right) \quad (26)$$

$$Q(t+1) = C_1 Q(t) + C_2 Q_{\text{self}}(t) + C_3 Q_{\text{global}}(t).$$

The quantum method has been applied to a swarm optimization algorithm in combinatorial optimization [109], cooperative approach [110], knapsack problems [108], and power quality monitor [111]. In differential evolution, it has been applied to knapsack problems [112], combinatorial problems [113], and image threshold methods [114]. The cuckoo search metaheuristic has been used for 0-1 knapsack problems [115] and the bin packing problem [116]. In ant colony optimization, it has been applied to image threshold [114]. The harmony search [117] and the Monkey algorithms [118] were applied to Knapsack problems.

Binary Method Based on Estimation of Distribution. Estimation of distribution algorithms (EDA) belong to promising regions. They are probabilistic models used in optimization methods. These methods guide the search for the optimum by sampling the promising candidates and building a distribution [119].

The new solutions are obtained by sampling the search space using EDA. After each iteration, the distribution is reestimated using the new candidates.

In the case of binary optimization, [120] used a univariate marginal distribution (UMD) to obtain a binary method.

Let n be the space dimension, N be the number of candidates, P_i be the best position for the particle, and P_g be the global best position. P_i and P_g are binary variables.

We want to obtain a particle $T = (T_1, T_2, \ldots, T_n)$, where T_i is the probability that the ith dimension of a solution takes the value 1. Let d be a specific dimension; to initialize the particle T, we apply the rule

$$T^d = \frac{\sum_{i=1}^{N} P_i^d}{N} \quad (27)$$

With the particle T, for an element x_i, where $i \in (1, \ldots, N)$, we apply the next decision:

If random() $< \beta$, then

$$x_i^d(t+1) = \begin{cases} 1 & \text{if random}() < T_d \\ 0, & \text{otherwise.} \end{cases} \quad (28)$$

Else

$$x_i^d(t+1) = P_g^d(t). \quad (29)$$

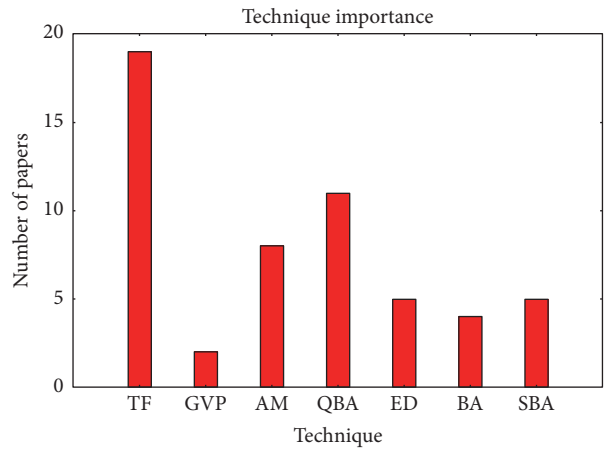

Figure 6: Number of papers by technique.

With this rule, we obtain $x_i^d(t+1), \forall d \in (1,\ldots,n)$. The next step is to update T particle. We use the rule

$$T^d = (1-\lambda)\,T^d + \lambda\frac{\sum_{i=1}^{N} P_i^d}{N}. \tag{30}$$

This method was constructed for a particle swarm optimization technique. However, it is easy to adapt for other metaheuristics. The advantage of this procedure is its adaptation on each iteration; however, it needs to adjust the parameters λ and β and to compute the distribution in each iteration. In PSO, it was applied to solve knapsack problems [120]. In differential evolution, it was applied to optimization problems [121, 122]. For genetic algorithms, it was used to work on economic dispatch problems [106]. Finally, local search [123] and memetic [123] metaheuristics were used to solve the traveling salesman problem.

6. Discussion

This section aims to summarize the techniques and problems recently addressed. Additionally with the information obtained from the articles along with our experience in the area, we want to capture our vision of what are the trends in binarization. This last point is very difficult to answer and is not intended to be a quantitative analysis but rather our vision regarding the area.

From 65 papers, we have summarized, reviewed, and classified techniques that allow transforming continuous metaheuristics into discrete or binary metaheuristics. Figure 6 shows how the articles are distributed on the different binarization techniques. The most reviewed technique was the transfer function. From the 19 read articles about this technique, it is observed that there is a general and simple mechanism for performing binarizations. However, the results are not always suitable and are related to the choice of the transfer function. In this sense the greatest challenge, in our view, corresponds to developing a methodology for choosing the transfer function (not simply trial and error) where this selection could be dynamic as the system evolves.

Another technique that appeared quite often was the quantum binary approach (QBA). From the articles read it is observed that the implementations are particular for each metaheuristic, with quite good results. From our point of view, there is a line of research associated with designing a general quantum mechanism that allows binarizing any continuous metaheuristic. Another important point to work on is the development of a methodology for the selection of parameters associated with binarization.

For the case of angle modulation, there is space to perform binarizations on new metaheuristics, where different variations of angle modulation can be explored. This exploration of new binarizations is very powerful if it is accompanied by analysis of positions and velocities of particles of the system to understand the conditions in which angle modulation works properly. As a suggestion we propose reading the work done in [124] in PSO.

From the point of view of problems, the greater number of problems addressed corresponds to classic problems such as the knapsack (KS), set covering problem (SCP), and traveling salesman problem (TSP). The summary is shown in Figure 7. With the generation of large amounts of data and the incorporation of the Internet of things, there is a large space to use metaheuristics associated with combinatorial problems in the area of image processing and feature selection [125], deep learning tuning parameters [126], data intensive applications [127–130], and network and complex systems [131]. In this context, feature selection (FS) has been resolved using angle modulation and the set-based approach. Image thresholding (IT) has been addressed using the quantum binary approach.

7. Conclusion

This work surveyed important discretization and binarization methods of continuous metaheuristics. Inside the binarization conglomerate, we propose two main group classifications. The first group we call two-step binarization methods, which use an intermediate space from where the binarization is mapped. The second group we call continuous-binary

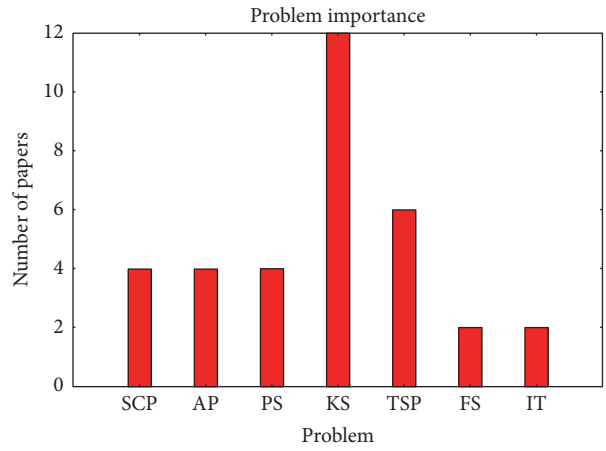

Figure 7: Number of papers by problem.

Table 3: Classification summary of binary approach.

Discretization	(i) Rounding off	
	(ii) Random-key or small value position	
	(iii) Metaheuristic discretization	
Binarization	(i) Two-step binarization	(i) Transfer function and binarization
		(ii) Great value priority (GVP) and mapping
		(iii) Angle modulation and rule
	(ii) Continuous-binary operator transformation	(i) Modified algebraic operations: set-based approach
		(ii) Promising regions: quantum binary approach
		(iii) Promising regions: binary method based on estimation of distribution

Table 4: Summary of discretization methods.

Binarization techniques	Metaheuristic	Problem	References
Rounding off	Particle swarm	Voltage control	[65]
		Transport aircraft wing	[64]
		Task assignment	[62]
	Ant colony	Distribution system Reconfiguration	[63]
	Firefly	Integer programming	[66]
Random-key or small value position	Memetic	Shop scheduling	[68]
	Hybrid gravitational-annealing	Traveling salesman	[69]
	Particle swarm	Permutation flow-shop sequencing	[70]
	Firefly	Scheduling jobs on grid computing	[71]
	Cuckoo search	Reliable embedded system reconfiguration	[72]
Metaheuristic discretizations	Teaching-learning	Distribution system	[73]

operator transformation, where the metaheuristic operator is adapted to a binary problem. When we analyze the operator adaptation, we found methods that transform the algebraic operations and methods that use a probability for performing the transition in the search space. Table 3 summarizes these results.

Moreover, we provide a summary of the main discretization techniques, indicating the metaheuristic that was used and what problem was resolved. This summary is shown in Table 4.

Additionally, we investigate what specific metaheuristics use these binarization techniques. The conclusion is that the

Table 5: Summary of two-step binarization methods.

Binarization techniques	Metaheuristic	Problem	References
Transfer function	Firefly	Set covering problem	[132, 133]
		Synthesis of thinned planar antenna array	[134]
		Nonlinear binary optimization	[135]
		Network and reliability constrained unit commitment problem	[78]
		Permutation flow-shop scheduling problem	[136]
	Algae	Knapsack problem	[41]
	Artificial bee colony	Set covering problem	[96]
		Thermal unit commitment	[137]
	Cuckoo search	Bulk power system	[37]
	Differential evolutionary	Multiagent systems	[29]
		Knapsack problems	[30]
	Binary bat	Unimodal, multimodal	[138]
		Traveling Salesman	[139]
	Gravitational search	Unimodal, multimodal	[80, 138]
	Open source development model	Combinatorial problems	[140]
	Particle swarm	Optimize sizing of capacitor banks	[82]
		Bulk power system	[81]
		Network reconfiguration	[83]
		Unit commitment problem	[31]
		Knapsack problems	[41, 74]
	Teaching-learning based	Designing plasmonic nanobipyramids based on absorption coefficient	[85]
	Electromagnetism-like method	Traveling sales	[141]
	Catfish	Feature selection	[142]
Great value priority	Binary Bat	Antenna positioning problem	[87]
	Particle swarm	Quadratic assignment problem	[86]
Angle modulation	Particle swarm	N-queens	[92]
		Binary problems	[89]
		Finding defensive islands of large-scale power systems	[90]
	Differential evolution	Knapsack problems	[93]
		Binary problems	[94]
	Artificial bee colony	Binary problems	[96]
		Feature selection	[95]
	Binary bat	Graph coloring	[143]
		Antenna positioning problem	[87]

most frequently used method is the transfer function, belonging to two-step binarization. Furthermore, we searched for what type of optimizations problems were resolved by the different techniques. The summary is shown in Tables 5 and 6.

The principal research in this area is to try to understand in a general way how exploration and exploitation properties are mapped from continuous metaheuristics to discrete or binary metaheuristics. This allows improving the result of the metaheuristics and enlarging the spectrum of discrete or binary problems to solve. This compilation work of discretization and binarization techniques allows us to conclude that no general technique exists that allows for efficient discretization.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Broderick Crawford is supported by Grant CONICYT/FONDECYT/REGULAR/1171243, Ricardo Soto is supported

Table 6: Summary of continuous-binary operator transformation.

Binarization techniques	Metaheuristic	Problem	Reference
Boolean approach	Particle swarm	Antenna design problem	[98, 99]
		Binary problems	[97]
	Binary artificial bee colony	Binary problems	[100]
Set-based approach	Particle swarm	Traveling salesman problem	[103]
		Multidimensional knapsack problem	
		Vehicle routing problem	[101]
	Jumping frogs	Feature selection	[144]
		Combinatorial problems	[145]
	Water cycle	Truss structure	[146]
	Mine blast	Truss structure	[146]
	Gravitational	Traveling Salesman	[147]
	Imperialist competition	Transmission expansion Planning	[148]
	Invasive weed	Typical benchmark functions (Sphere, Rosenbrock, Rastrigin, Griewank)	[149]
	Social impact theory	Pattern recognition	[150]
Quantum binary approach	Particle swarm	Competitive facility	[151]
		Location problems	
		Knapsack problem	[108]
		Power quality monitor placement method	[111]
	Differential evolution	Knapsack problem	[151]
		Combinatorial problems	[113]
		Image thresholding	[114]
	Cuckoo search	0-1 knapsack problem	[115]
		Bin packing problem	[116]
	Ant colony optimization	Image thresholding	[114]
	Harmony search	0-1 knapsack problem	[117]
	Monkey	0-1 knapsack problem	[118]
Binary method based on estimation of distribution	Particle swarm	Knapsack problem	[120]
	Differential Evolution	Optimization problems	[121, 122]
	Genetic	Economic dispatch problem	[106]
	Local search	Probabilistic traveling salesman problem	[123]
	Memetic	Probabilistic traveling salesman problem	[123]

by Grant CONICYT/FONDECYT/REGULAR/1160455, José García is supported by INF-PUCV 2016, and Gino Astorga is supported by Postgraduate Grant, Pontificia Universidad Católica de Valparaíso, 2015.

References

[1] A. Abshouri, M. Meybodi, and A. Bakhtiary, "New firefly algorithm based on multi swarm learning automata in dynamic environments," in *Proceedings of IEEE*, vol. 13, pp. 989–993, 2011.

[2] S. Kazemzadeh Azad, "Optimum design of structures using an improved firefly algorithm," *Iran University of Science & Technology*, vol. 1, no. 2, pp. 327–340, 2011.

[3] T. Apostolopoulos and A. Vlachos, "Application of the firefly algorithm for solving the economic emissions load dispatch problem," *International Journal of Combinatorics*, vol. 2011:999, 2010.

[4] J. C. Bansal and K. Deep, "Optimization of directional overcurrent relay times by particle swarm optimization," in *Proceedings of the 2008 IEEE Swarm Intelligence Symposium (SIS '08)*, pp. 1–7, St. Louis, Mo, USA, September 2008.

[5] O. Bénichou, C. Loverdo, M. Moreau, and R. Voituriez, "Two-dimensional intermittent search processes: an alternative to lévy flight strategies," *Physical Review E*, vol. 74, no. 2, Article ID 020102, 2006.

[6] S. Nesmachnow, "An overview of metaheuristics: accurate and efficient methods for optimisation," *International Journal of Metaheuristics*, vol. 3, no. 4, pp. 320–347, 2014.

[7] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Courier Corporation, 2001.

[8] M. Grotschel and L. Lovasz, "Combinatorial optimization," *Handbook of Combinatorics*, vol. 2, no. 168, pp. 1541–1597, 1995.

[9] E. Talbi, *Metaheuristics - From Design to Implementation*, Wiley, 2009.

[10] R. L. Graham, *Handbook of Combinatorics*, vol. 1, Elsevier, 1995.

[11] G. L. Nemhauser and L. A. Wolsey, *Integer Programming and Combinatorial Optimization*, Wiley, 1992.

[12] G. L. Nemhauser, M. W. P. Savelsbergh, and G. S. Sigismondi, "Constraint classification for mixed integer programming formulations," *Coal Bulletin*, vol. 20, pp. 8–12, 1988.

[13] F. Neumann and C. Witt, "Bioinspired computation in combinatorial optimization: algorithms and their computational complexity," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation (GECCO '13)*, pp. 567–590, The Netherlands, Companion Material Proceedings, July 6-10, 2013.

[14] C. Blum and A. Roli, "Metaheuristics in combinatorial optimi zation: overview and conceptual comparison," *ACM Computing Surveys*, vol. 35, no. 3, pp. 268–308, 2003.

[15] F. Glover and G. Kochenberger, "The ant colony optimization metaheuristic: algorithms, applications, and advances," *Handbook of Metaheuristics*, pp. 250–285, 2003.

[16] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of machine learning*, pp. 760–766, Springer, 2011.

[17] S. Mirjalili and A. S. Sadiq, "Magnetic Optimization Algorithm for training Multi Layer Perceptron," in *Proceedings of the IEEE 3rd International Conference on Communication Software and Networks (ICCSN '11)*, pp. 42–46, May 2011.

[18] X.-S. Yang and S. Deb, "Cuckoo search via lévy flights," in *Proceedings of the Nature & Biologically Inspired Computing (NaBIC '09) World Congress*, pp. 210–214, IEEE, 2009.

[19] X. Yang, "Firefly algorithm, stochastic test functions and design optimization," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78–84, 2010.

[20] H. Shah-Hosseini, "Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation," *International Journal of Computational Science and Engineering*, vol. 6, no. 1-2, pp. 132–140, 2011.

[21] Y. Shi, "Brain storm optimization algorithm," in *Proceedings of the International Conference in Swarm Intelligence*, pp. 303–309, Springer, 2011.

[22] H. Shareef, A. A. Ibrahim, and A. H. Mutlag, "Lightning search algorithm," *Applied Soft Computing Journal*, vol. 36, pp. 315–333, 2015.

[23] S. Mirjalili, "Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, pp. 228–249, 2015.

[24] S. Mirjalili, "SCA: A Sine Cosine Algorithm for solving optimization problems," *Knowledge-Based Systems*, 2015.

[25] A. Hatamlou, "Black hole: a new heuristic optimization approach for data clustering," *Information Sciences*, vol. 222, pp. 175–184, 2013.

[26] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[27] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[28] T. Gong and A. L. Tuson, "Differential evolution for binary encoding," in *Soft Computing in Industrial Applications*, pp. 251–262, Springer, 2007.

[29] W. Zhifeng, H. Houkuan, and Z. Xiang, "A binary-encoding differential evolution algorithm for agent coalition," *Journal of Computer Research and Development*, vol. 5, no. 019, 2008.

[30] H.-Y. Cai, Z.-F. Hao, Z.-G. Wang, and G. Guo, "Binary differential evolution algorithm for 0-1 knapsack problem," *Computer Engineering and Design*, vol. 7:047, 2009.

[31] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proceedings of the Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference*, vol. 5, pp. 4104–4108, IEEE, 1997.

[32] M. H. Tayarani and N. M. R. Akbarzadeh. T., "Magnetic optimization algorithms a new synthesis," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '08)*, pp. 2659–2664, Hong Kong, China, June 1-6, 2008.

[33] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 213, pp. 267–289, 2010.

[34] B. Crawford, R. Soto, M. Olivares-Suárez, and F. Paredes, "A binary firefly algorithm for the set covering problem," *Advances in Intelligent Systems and Computing*, vol. 285, pp. 65–73, 2014.

[35] B. Crawford, R. Soto, C. Peña et al., "Binarization methods for shuffled frog leaping algorithms that solve set covering problems," *Advances in Intelligent Systems and Computing*, vol. 349, pp. 317–326, 2015.

[36] B. Crawford, R. Soto, C. Torres-Rojas et al., "A binary fruit fly optimization algorithm to solve the set covering problem," in *Proceedings of the International Conference on Computational Science and Its Applications*, pp. 411–420, Springer, 2015.

[37] R. Soto, B. Crawford, R. Olivares, J. Barraza, F. Johnson, and F. Paredes, "A binary cuckoo search algorithm for solving the set covering problem," in *Proceedings of the International Work-Conference on the Interplay Between Natural and Artificial Computation*, pp. 88–97, Springer, 2015.

[38] B. Crawford, R. Soto, N. Berrios, and E. Olguin, "Solving the set covering problem using the binary cat swarm optimization metaheuristic," *World Academy of Science, Engineering and Technology, International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering*, vol. 10, no. 3, pp. 104–108, 2016.

[39] B. Crawford, R. Soto, C. Olea, F. Johnson, and F. Paredes, "Binary bat algorithms for the set covering problem," in *Proceedings of the 10th Iberian Conference on Information Systems and Technologies (CISTI '15)*, prt, June 2015.

[40] J. García, B. Crawford, R. Soto, and P. García, "A multi dynamic binary black hole algorithm applied to set covering problem," in *Proccedings of the International Conference on Harmony Search Algorithm*, pp. 42–51, Springer, 2017.

[41] X. Zhang, C. Wu, J. Li et al., "Binary artificial algae algorithm for multidimensional knapsack problems," *Applied Soft Computing*, vol. 43, pp. 583–595, 2016.

[42] K. S. Reddy, L. K. Panwar, R. Kumar, and B. K. Panigrahi, "Binary fireworks algorithm for profit based unit commitment (PBUC) problem," *International Journal of Electrical Power & Energy Systems*, vol. 83, pp. 270–282, 2016.

[43] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1155–1173, 2008.

[44] P. Guo and L. Zhu, "Ant colony optimization for continuous domains," in *Proceedings of the 8th International Conference on Natural Computation (ICNC '12)*, pp. 758–762, Chongqing, China, May 2012.

[45] P. A. Bosman and D. Thierens, "Continuous iterated density estimation evolutionary algorithms within the idea framework," 2001.

[46] G. E. Box, M. E. Muller et al., "A note on the generation of random normal deviates," *The Annals of Mathematical Statistics*, vol. 29, no. 2, pp. 610–611, 1958.

[47] A. Eiben and J. Smith, *Introduction to Evolutionary Computing*, Natural Computing Series, Springer-Verlag, Berlin, Germany, 2nd edition, 2015.

[48] S. L. Tilahun and J. T. Ngnotchouye, "Firefly algorithm for optimization problems with non-continuous variables: a review and analysis," *Computing Research Repository*, 2016, https://arxiv.org/abs/1602.07884.

[49] M. Grötschel, "Discrete mathematics in manufacturing," *ICIAM*, vol. 91, pp. 119–145, 1992.

[50] H. Dyckhoff, "A typology of cutting and packing problems," *European Journal of Operational Research*, vol. 44, no. 2, pp. 145–159, 1990.

[51] L. A. Wolsey, *Integer Programming*, vol. 42, John Wiley & Sons, New York, NY, USA, 1998.

[52] P. Pardalos, H. Wolkowicz et al., "Quadratic Assignment and Related Problems: DIMACS Workshop," May 20-21, 1993, volume 16. American Mathematical Soc., 1994.

[53] J. Lv, X. Wang, M. Huang, H. Cheng, and F. Li, "Solving 0-1 knapsack problem by greedy degree and expectation efficiency," *Applied Soft Computing Journal*, vol. 41, pp. 94–103, 2016.

[54] B. Crawford, R. Soto, M. Olivares-Suárez et al., "A binary coded firefly algorithm that solves the set covering problem," *Romanian Journal of Information Science and Technology*, vol. 17, no. 3, pp. 252–264, 2014.

[55] E. Osaba, X.-S. Yang, F. Diaz, P. Lopez-Garcia, and R. Carballedo, "An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems," *Engineering Applications of Artificial Intelligence*, vol. 48, pp. 59–71, 2016.

[56] I. H. Osman and G. Laporte, "Metaheuristics: a bibliography," *Annals of Operations Research*, vol. 63, pp. 513–623, 1996.

[57] C. Blum, "Ant colony optimization," in *Proceedings of the 13th Annual Genetic and Evolutionary Computation Conference (GECCO '11)*, pp. 963–990, Companion Material Proceedings, Dublin, Ireland, July 2011.

[58] Ö. Babaoglu, T. Binci, M. Jelasity, and A. Montresor, "Firefly-inspired heartbeat synchronization in overlay networks," in *Proceedings of the First International Conference on Self-Adaptive and Self-Organizing Systems (SASO '07)*, pp. 77–86, Boston, Mass, USA, July 2007.

[59] X. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, 2010.

[60] S. L. Tilahun and H. C. Ong, "Prey-predator algorithm: A new metaheuristic algorithm for optimization problems," *International Journal of Information Technology and Decision Making*, vol. 14, no. 6, pp. 1331–1352, 2015.

[61] B. Yuce, M. S. Packianather, E. Mastrocinque, D. T. Pham, and A. Lambiase, "Honey bees inspired optimization method: the bees algorithm," *Insects*, vol. 4, no. 4, pp. 646–662, 2013.

[62] A. Salman, I. Ahmad, and S. Al-Madani, "Particle swarm optimization for task assignment problem," *Microprocessors and Microsystems*, vol. 26, no. 8, pp. 363–371, 2002.

[63] A. Y. Abdelaziz, R. A. Osama, S. M. El-Khodary, and B. K. Panigrahi, *Distribution Systems Reconfiguration Using the Hyper-Cube Ant Colony Optimization Algorithm*, Springer Berlin Heidelberg, Berlin, Germany, 2011.

[64] G. Venter and J. Sobieszczanski-Sobieski, "Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization," *Structural and Multidisciplinary Optimization*, vol. 26, no. 1-2, pp. 121–131, 2004.

[65] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A Particle swarm optimization for reactive power and voltage control considering voltage security assessment," *IEEE Transactions on Power Systems*, vol. 15, no. 4, pp. 1232–1239, 2000.

[66] N. Bacanin, I. Brajevic, and M. Tuba, "Firefly algorithm applied to integer programming problems," *Recent Advances in Mathematics*, vol. 888:999, 2013.

[67] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," in *Proceedings of the Foundations of Fuzzy Logic and Soft Computing, 12th International Fuzzy Systems Association World Congress, (IFSA '07)*, pp. 789–798, Cancun , Mexico, June 18-21, 2007.

[68] X. Li, J. Wang, J. Zhou, and M. Yin, "An effective GSA based memetic algorithm for permutation flow shop scheduling," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '10)*, pp. 1–6, Barcelona, Spain, 2010.

[69] H. Chen, S. Li, and T. Zheng, "Hybrid gravitational search algorithm with random-key encoding scheme combined with simulated annealing," *International Journal of Computer Science and Network Security*, vol. 11, no. 6, pp. 208–217, 2011.

[70] M. F. Tasgetiren, Y. C. Liang, M. Sevkli, and G. A. Gencyilmaz, "A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem," *European Journal of Operational Research*, vol. 177, no. 3, pp. 1930–1947, 2007.

[71] A. Yousif, A. H. Abdullah, S. M. Nor, and A. A. Abdelaziz, "Scheduling jobs on grid computing using firefly algorithm," *Journal of Theoretical and Applied Information Technology*, vol. 33, no. 2, pp. 155–164, 2011.

[72] A. Kumar and S. Chakarverty, "Design optimization for reliable embedded system using Cuckoo search," in *Proceedings of the 3rd International Conference on Electronics Computer Technology (ICECT '11)*, vol. 1, pp. 264–268, April 2011.

[73] A. Lotfipour and H. Afrakhte, "A discrete teaching-learning-based optimization algorithm to solve distribution system reconfiguration in presence of distributed generation," *International Journal of Electrical Power and Energy Systems*, vol. 82, pp. 264–273, 2016.

[74] S. Palit, S. N. Sinha, M. A. Molla, A. Khanra, and M. Kule, "A cryptanalytic attack on the knapsack cryptosystem using binary firefly algorithm," in *Proceedings of the International Conference on Computer and Communication Technology (ICCCT '11)*, vol. 2, pp. 428–432, 2011.

[75] M. K. Sayadi, A. Hafezalkotob, and S. G. J. Naini, "Firefly-inspired algorithm for discrete optimization problems: an application to manufacturing cell formation," *Journal of Manufacturing Systems*, vol. 32, no. 1, pp. 78–84, 2013.

[76] N. Rajalakshmi, P. D. Subramanian, and K. Thamizhavel, "Performance enhancement of radial distributed system with distributed generators by reconfiguration using binary firefly algorithm," *Journal of The Institution of Engineers (India)*, vol. 96, no. 1, Series B, pp. 91–99, 2015.

[77] Y. Yang, Y. Mao, P. Yang, and Y. Jiang, "The unit commitment problem based on an improved firefly and particle swarm optimization hybrid algorithm," in *Proceedings of the Chinese Automation Congress (CAC '13)*, pp. 718–722, IEEE, 2013.

[78] K. Chandrasekaran and S. P. Simon, "Network and reliability constrained unit commitment problem using binary real coded firefly algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 43, no. 1, pp. 921–932, 2012.

[79] K. Chandrasekaran, S. P. Simon, and N. P. Padhy, "Binary real coded firefly algorithm for solving unit commitment problem," *Information Sciences*, vol. 249, pp. 67–84, 2013.

[80] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "BGSA: binary gravitational search algorithm," *Natural Computing*, vol. 9, no. 3, pp. 727–745, 2010.

[81] T. Khalil, H. Youseef, and M. Aziz, "A binary particle swarm optimization for optimal placement and sizing of capacitor banks in radial distribution feeders with distorted substation voltages," in *Proceedings of the AIML international conference*, pp. 137–143, 2006.

[82] D. Robinson, "Reliability analysis of bulk power systems using swarm intelligence," *IEEE*, pp. 96–102, 2005.

[83] Y. Liu and X. Gu, "Skeleton-network reconfiguration based on topological characteristics of scale-free networks and discrete particle swarm optimization," *IEEE Transactions on Power Systems*, vol. 22, no. 3, pp. 1267–1274, 2007.

[84] B. Crawford, R. Soto, R. Cuesta, M. Olivares-Suárez, F. Johnson, and E. Olgun, "Two swarm intelligence algorithms for the set covering problem," in *Proceedings of the 9th International Conference on Software Engineering and Applications, (ICSOFT-EA '14)*, pp. 29–31, Vienna, Austria, 29-31 August, 2014.

[85] M. Akhlaghi, F. Emami, and N. Nozhat, "Binary tlbo algorithm assisted for designing plasmonic nano bi-pyramids-based absorption coefficient," *Journal of Modern Optics*, vol. 61, no. 13, pp. 1092–1096, 2014.

[86] C. Lv, H. Zhao, and X. Yang, "Particle swarm optimization algorithm for quadratic assignment problem," in *Proceedings of the International Conference on Computer Science and Network Technology (ICCSNT '11)*, pp. 1728–1731, December 2011.

[87] Z. A. E. Moiz Dahi, C. Mezioud, and A. Draa, "Binary bat algorithm: On the efficiency of mapping functions when handling binary problems using continuous-variable-based metaheuristics," *IFIP Advances in Information and Communication Technology*, vol. 456, pp. 3–14, 2015.

[88] J. Proakis and M. Salehi, *Communication Systems Engineering*, vol. 3, Prentice Hall, Upper Saddle River, NJ, USA, 2 edition, 2002.

[89] G. Pampara, N. Franken, and A. P. Engelbrecht, "Combining particle swarm optimisation with angle modulation to solve binary problems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '05)*, vol. 1, pp. 89–96, Edinburgh, UK, September 2005.

[90] W. Liu, L. Liu, and D. Cartes, "Angle modulated particle swarm optimization based defensive islanding of large scale power systems," in *Proceedings of the IEEE Power Engineering Society Conference and Exposition in Africa*, pp. 1–8, 2007.

[91] S. Das, R. Mukherjee, R. Kundu, and T. Vasilakos, "Multi-user detection in multi-carrier CDMA wireless broadband system using a binary adaptive differential evolution algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '13)*, pp. 1245–1252, Amsterdam, The Netherlands, July 6-10, 2013.

[92] B. J. Leonard and A. P. Engelbrecht, "Frequency distribution of candidate solutions in angle modulated particle swarms," in *Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI '15)*, pp. 251–258, Cape Town, South Africa, December 7-10, 2015.

[93] C. Deng, T. Weise, and B. Zhao, "Pseudo binary differential evolution algorithm," *Journal of Computer Information Systems*, vol. 8, no. 6, pp. 2425–2436, 2012.

[94] G. Pampará, A. P. Engelbrecht, and N. Franken, "Binary differential evolution," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 1873–1879, Vancouver, BC, Canada, July 2006, Part of WCCI 2006.

[95] G. Yavuz and D. Aydin, "Angle modulated artificial bee colony algorithms for feature selection," *Applied Computational Intelligence and Soft Computing*, vol. 2016, Article ID 9569161, 6 pages, 2016.

[96] G. Pampará and A. P. Engelbrecht, "Binary artificial bee colony optimization," in *Proceedings of the IEEE Symposium on Swarm Intelligence (SIS '11)*, pp. 1–8, IEEE Perth, April 2011.

[97] F. Afshinmanesh, A. Marandi, and A. Rahimi-Kian, "A novel binary particle swarm optimization method using artificial immune system," in *Proceedings of the International Conference on Computer as a Tool (EUROCON '05)*, vol. 1, pp. 217–220, IEEE, 2005.

[98] A. Marandi, F. Afshinmanesh, M. Shahabadi, and F. Bahrami, "Boolean particle swarm optimization and its application to the design of a dual-band dual-polarized planar antenna," in *Proceedings of the IEEE International Conference on Evolutionary Computation (CEC '06)*, pp. 3212–3218, Vancouver, BC, Canada, July, 2006, Part of WCCI 2006.

[99] K. V. Deligkaris, Z. D. Zaharis, D. G. Kampitaki, S. K. Goudos, I. T. Rekanos, and M. N. Spasos, "Thinned planar array design using boolean PSO with velocity mutation," *IEEE Transactions on Magnetics*, vol. 45, no. 3, pp. 1490–1493, 2009.

[100] D. Jia, X. Duan, and M. K. Khan, "Binary artificial bee colony optimization using bitwise operation," *Computers and Industrial Engineering*, vol. 76, pp. 360–365, 2014.

[101] Y. J. Gong, J. Zhang, O. Liu, R. Z. Huang, H. S. H. Chung, and Y.-H. Shi, "Optimizing the vehicle routing problem with time windows: a discrete particle swarm optimization approach," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 42, no. 2, pp. 254–267, 2012.

[102] M. Neethling and A. P. Engelbrecht, "Determining RNA secondary structure using set-based particle swarm optimization," in *Proceedings of the IEEE International Conference on Evolutionary Computation (CEC '06)*, pp. 1670–1677, Vancouver , BC, Canada, July 2006, Part of WCCI 2006.

[103] W. N. Chen, J. Zhang, H. S. H. Chung, W. L. Zhong, W. G. Wu, and Y. H. Shi, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 2, pp. 278–300, 2010.

[104] J. Langeveld and A. Engelbrecht, "A generic set-based particle swarm optimization algorithm," in *Proceedings of the International Conference on Swarm Intelligence (ICSI '11)*, Paris, France, 2011.

[105] J. Langeveld and A. P. Engelbrecht, "Set-based particle swarm optimization applied to the multidimensional knapsack problem," *Swarm Intelligence*, vol. 6, no. 4, pp. 297–342, 2012.

[106] Y.-P. Chen and C.-H. Chen, "Enabling the extended compact genetic algorithm for real-parameter optimization by using adaptive discretization," *Evolutionary Computation*, vol. 18, no. 2, pp. 199–228, 2010.

[107] J. Sun, B. Feng, and W. Xu, "Particle swarm optimization with particles having quantum behavior," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '04)*, pp. 325–331, Portland, Ore, USA, June 2004.

[108] Y. Shuyuan, W. Min, and J. Licheng, "A quantum particle swarm optimization," *IEEE Congress on Evolutionary Computation*, vol. 1, pp. 19–23, 2004.

[109] J. Wang, Y. Zhang, Y. Zhou, and J. Yin, "Discrete quantum-behaved particle swarm optimization based on estimation of distribution for combinatorial optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '08)*, pp. 897–904, Hong Kong, China, June 1-6, 2008.

[110] J. Zhao, J. Sun, and W. Xu, "A binary quantum-behaved particle swarm optimization algorithm with cooperative approach," *International Journal of Computer Science*, vol. 10, no. 2, pp. 112–118, 2005.

[111] A. A. Ibrahim, A. Mohamed, H. Shareef, and S. P. Ghoshal, "An effective power quality monitor placement method utilizing quantum-inspired particle swarm optimization," in *Proceedings of the International Conference on Electrical Engineering and Informatics (ICEEI '11)*, pp. 1–6, Bandung, Indonesia, 7-19 July, 2011.

[112] A. R. Hota and A. Pat, "An adaptive quantum-inspired differential evolution algorithm for 0-1 knapsack problem," in *Proceedings of the 2nd World Congress on Nature and Biologically Inspired Computing (NaBIC '10)*, pp. 703–708, December 2010.

[113] J. Alegra and Y. Túpac, "A generalized quantum-inspired evolutionary algorithm for combinatorial optimization problems," in *Proceedings of the XXXII International Conference of the Chilean Computer Science Society (SCCC '14)*, pp. 11–15, November.

[114] S. Dey, S. Bhattacharyya, and U. Maulik, "New quantum inspired meta-heuristic techniques for multi-level colour image thresholding," *Applied Soft Computing*, vol. 46, pp. 677–702, 2016.

[115] A. Layeb, "A Novel Quantum Inspired Cuckoo Search for Knapsack Problems," *International Journal of Bio-Inspired Computation*, vol. 3, no. 5, pp. 297–305, 2011.

[116] A. Layeb and S. R. Boussalia, "A novel quantum inspired cuckoo search algorithm for bin packing problem," *International Journal of Information Technology and Computer Science*, vol. 4, no. 5, 58 pages, 2012.

[117] A. Layeb, "A hybrid quantum inspired harmony search algorithm for 0-1 optimization problems," *Journal of Computational and Applied Mathematics*, vol. 253, pp. 14–25, 2013.

[118] Y. Zhou, X. Chen, and G. Zhou, "An improved monkey algorithm for a 0-1 knapsack problem," *Applied Soft Computing Journal*, vol. 38, pp. 817–830, 2016.

[119] P. Larranaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Kluwer Academic Publishers, Boston, UK, 2nd edition, 2002.

[120] J. Wang, "A novel discrete particle swarm optimization based on estimation of distribution," in *Proceedings of the Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence, Third International Conference on Intelligent Computing (ICIC '07)*, pp. 791–802, Qingdao, China, August 21-24, 2007.

[121] S. Tsutsui, M. Pelikan, and D. E. Goldberg, "Evolutionary algorithm using marginal histogram models in continuous domain," *IlliGAL Report*, vol. 2001019, 2001.

[122] M. Pelikan, D. E. Goldberg, and S. Tsutsui, "Getting the best of both worlds: discrete and continuous genetic and evolutionary algorithms in concert," *Information Sciences*, vol. 156, no. 3-4, pp. 147–171, 2003.

[123] P. Balaprakash, M. Birattari, T. Stützle, and M. Dorigo, "Estimation-based metaheuristics for the probabilistic traveling salesman problem," *Computers & OR*, vol. 37, no. 11, pp. 1939–1951, 2010.

[124] B. J. Leonard, A. P. Engelbrecht, and C. W. Cleghorn, "Critical considerations on angle modulated particle swarm optimisers," *Swarm Intelligence*, vol. 9, no. 4, pp. 291–314, 2015.

[125] F. Barani, M. Mirhosseini, and H. Nezamabadi-pour, "Application of binary quantum-inspired gravitational search algorithm in feature subset selection," *Applied Intelligence*, pp. 1–15, 2017.

[126] J.-S. Chou and J. P. P. Thedja, "Metaheuristic optimization within machine learning-based classification system for early warnings related to geotechnical problems," *Automation in Construction*, vol. 68, pp. 65–80, 2016.

[127] K.-C. Lin, K.-Y. Zhang, Y.-H. Huang, J. C. Hung, and N. Yen, "Feature selection based on an improved cat swarm optimization algorithm for big data classification," *Journal of Supercomputing*, pp. 1–12, 2016.

[128] L. Shang, Z. Zhou, and X. Liu, "Particle swarm optimization-based feature selection in sentiment classification," *Soft Computing*, vol. 20, no. 10, pp. 3821–3834, 2016.

[129] L. Shen, H. Chen, Z. Yu et al., "Evolving support vector machines using fruit fly optimization for medical data classification," *Knowledge-Based Systems*, vol. 96, pp. 61–75, 2016.

[130] B. Z. Dadaneh, H. Y. Markid, and A. Zakerolhosseini, "Unsupervised probabilistic feature selection using ant colony optimization," *Expert Systems with Applications*, vol. 53, pp. 27–42, 2016.

[131] G. J. Krishna and V. Ravi, "Modified harmony search applied to reliability optimization of complex systems," *Advances in Intelligent Systems and Computing*, vol. 382, pp. 169–180, 2016.

[132] B. Crawford, R. Soto, M. O. Suárez, F. Paredes, and F. Johnson, "Binary Firefly algorithm for the set covering problem," in *Proceedings of the 9th Iberian Conference on Information Systems and Technologies (CISTI '14)*, pp. 1–5, June 2014.

[133] B. Crawford, R. Soto, M. Riquelme-Leiva et al., "Modified binary firefly algorithms with different transfer functions for solving set covering problems," *Advances in Intelligent Systems and Computing*, vol. 349, pp. 307–315, 2015.

[134] L. Pappula and D. Ghosh, "Synthesis of thinned planar antenna array using multiobjective normal mutated binary cat swarm optimization," *Applied Computational Intelligence and Soft Computing*, vol. 2016, Article ID 4102156, 9 pages, 2016.

[135] M. F. Costa, A. M. Rocha, R. B. Francisco, and E. M. Fernandes, "Heuristic-based firefly algorithm for bound constrained nonlinear binary optimization," *Advances in Operations Research*, Article ID 215182, 12 pages, 2014.

[136] M. K. Sayadi, R. Ramezanian, and N. Ghaffari-Nasab, "A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems," *International Journal of Industrial Engineering Computations*, vol. 1, no. 1, pp. 1–10, 2010.

[137] K. Chandrasekaran, S. Hemamalini, S. P. Simon, and N. P. Padhy, "Thermal unit commitment using binary/real coded artificial bee colony algorithm," *Electric Power Systems Research*, vol. 84, no. 1, pp. 109–119, 2012.

[138] S. Mirjalili, G.-G. Wang, and L. D. S. Coelho, "Binary optimization using hybrid particle swarm optimization and gravitational search algorithm," *Neural Computing and Applications*, vol. 25, no. 6, pp. 1423–1435, 2014.

[139] Y. Saji and M. E. Riffi, "A novel discrete bat algorithm for solving the travelling salesman problem," *Neural Computing and Applications*, vol. 27, no. 7, pp. 1853–1866, 2016.

[140] H. B. Khormouji, H. Hajipour, and H. Rostami, "BODMA: a novel metaheuristic algorithm for binary optimization problems based on open source development model algorithm," in

Proceedings of the 7th International Symposium on Telecommunications (IST '14), pp. 49–54, September 2014.

[141] N. Javadian, M. G. Alikhani, and R. Tavakkoli-Moghaddam, "A discrete binary version of the electromagnetism-like heuristic for solving traveling salesman problem," in *Proceedings of the Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence, 4th International Conference on Intelligent Computing, (ICIC '08)*, pp. 123–130, Shanghai , China, September 15-18, 2008.

[142] L.-Y. Chuang, S.-W. Tsai, and C.-H. Yang, "Catfish binary particle swarm optimization for feature selection," in *Proceedings of the International Conference on Machine Learning and Computing, (IPCSIT '11)*, vol. 3, pp. 40–44, 2011.

[143] H. Djelloul and S. Chikhi, "Combining bat algorithm with angle modulation for graph coloring problem," in *Proceedings of the Symposium on Complex Systems and Intelligent Computing, (Comp SIC '15)*, 2015.

[144] R. Jensen and Q. Shen, "Finding rough set reducts with ant colony optimization," in *Proceedings of the 2003 UK Workshop on Computational Intelligence*, vol. 1, 2003.

[145] F. J. M. Garcia and J. A. M. Perez, "Jumping frogs optimization: a new swarm method for discrete optimization," *Documentos de Trabajo del DEIOC*, vol. 3, 2008.

[146] A. Sadollah, H. Eskandar, A. Bahreininejad, and J. H. Kim, "Water cycle, mine blast and improved mine blast algorithms for discrete sizing optimization of truss structures," *Computers and Structures*, vol. 149, pp. 1–16, 2015.

[147] M. B. Dowlatshahi, H. Nezamabadi-pour, and M. Mashinchi, "A discrete gravitational search algorithm for solving combinatorial optimization problems," *Information Sciences*, vol. 258, pp. 94–107, 2014.

[148] E. A. Duki, H. A. Mansoorkhani, A. Soroudi, and M. Ehsan, "A discrete imperialist competition algorithm for transmission expansion planning," in *Proceedings of the 25th International Power System Conference*, pp. 1–10, 2010.

[149] C. Veenhuis, "Binary invasive weed optimization," in *Proceedings of the 2010 2nd World Congress on Nature and Biologically Inspired Computing (NaBIC '10)*, pp. 449–454, December 2010.

[150] M. Macaš, A. P. Bhondekar, R. Kumar et al., "Binary social impact theory based optimization and its applications in pattern recognition," *Neurocomputing*, vol. 132, pp. 85–96, 2014.

[151] S. A. MirHassani, S. Raeisi, and A. Rahmani, "Quantum binary particle swarm optimization-based algorithm for solving a class of bi-level competitive facility location problems," *Optimization Methods and Software*, vol. 30, no. 4, pp. 756–768, 2015.