A Survey of Swarm Algorithms Applied to Discrete Optimization Problems

Jonas Krause[a], Jelson Cordeiro[a], Rafael Stubs Parpinelli[a,b,*], Heitor Silvério Lopes[a]

[a]*Bioinformatics Laboratory, Federal University of Technology Paraná, Curitiba, Brazil*
[b]*Applied* Cognitive *Computing Group, Santa Catarina State University, Joinville, Brazil*

Abstract

Most swarm intelligence algorithms were devised for continuous optimization problems. However, they have been adapted for discrete optimization as well with applications in different domains. This survey aims at providing an updated review of research of swarm intelligence algorithms for discrete optimization problems, comprising combinatorial or binary. The biological inspiration that motivated the creation of each swarm algorithm is introduced, and later, the discretization and encoding methods used to adapt each algorithm for discrete problems. Methods are compared for different classes of problems and a critical analysis is provided, pointing to future trends.

Keywords: Swarm Intelligence, discrete domain, binary problems, combinatorial problems, bio-inspired algorithms

---

*Corresponding author. *Email addresses:* jkfries@gmail.com (Jonas Krause), jelsoncordeiro@gmail.com (Jelson Cordeiro), parpinelli@joinville.udesc.br (Rafael Stubs Parpinelli), hslopes@utfpr.edu.br (Heitor Silvério Lopes)

1. Introduction

Swarm-based algorithms are inspired by the behavior of some social living beings, such as ants, termites, birds, and fishes. Self-organization and decentralized control are remarkable features of swarm-based systems that, such as in nature, leads to an emergent behavior. Emergent behavior is a property that emerges through local interactions among system components and it is not possible to be achieved by any of the components of the system acting alone (Garnier et al., 2007).

In the beginning, the two mainstreams of the Swarm Intelligence area were Ant Colony Optimization (Dorigo and Stützle, 2004) and Particle Swarm Optimization (Kennedy and Eberhart, 2001). In recent years new swarm intelligence algorithms have appeared, inspired by fish schools (Cai, 2010), gravity and mass interactions (Rashedi et al., 2009), as well as different aspects of the behavior of bees (Abbass, 2001b; Karaboga, 2005; Lucic and Teodorovic, 2002; Pham et al., 2005), bacteria (Passino, 2002), glow-worms (Krishnanand and Ghose, 2005), fireflies (Yang, 2008), cockroaches (Havens et al., 2008), bats (Yang, 2009), and cuckoo birds (Yang and Deb, 2009). For a thorough review of recent approaches, see Parpinelli and Lopes (2011). Despite the swarm inspiration common to these approaches, they have their own particular way to exploit and explore the search space of the problem.

Although almost all the above cited algorithms were designed to be applied to continuous optimization, several of then were later adapted to handle discrete domain problems. Unlike the continuous domain, in which the elements have the property of varying smoothly, the elements inside a discrete domain – such as integers or binary digits – accept only distinct, separated values. The discrete domain is characterized by dealing with countable sets, either finite or infinite. Binary and combinatorial applications are examples of discrete domain problems.

This work reviews how swarm algorithms, that traditionally deal with continuous domains, can be

adapted to discrete problems. In this work we highlighted their adaptation strategies to handle this class of problems as well as their main features concerning the problem being solved. Notice that algorithms created specifically to handle discrete problems are beyond the scope of this work. This is the case, for instance, of the Ant Colony Optimization algorithm (Dorigo and Stützle, 2004), the Mosquito Host-seeking algorithm (Feng et al., 2009), the Calling Behavior of Japanese Tree Frogs algorithm (Hernández and Blum, 2012), the River Formation Dynamics algorithm (Rabanal et al., 2007), and the Intelligent Water Drops algorithm (Shah-Hosseini, 2007).

Section 2 shortly describes the bio-inspirations of the main swarm algorithms adapted to discrete problems; Section 3 brings the main concerns to handle discrete problems, the discretization methods and the encoding strategies; Section 4 details all the problems and applications addressed by each continuous algorithm; Section 5 summarizes and discusses of the main issues; and Section 6 presents the concluding remarks and points future research directions.

2. Swarm Algorithms

For all traditional versions of the algorithms discussed in this survey, in their essence, the candidate solutions are encoded as a set of real variables, which represent a point in a multidimensional space. In this section we briefly describe the swarm algorithms that are applied or adapted to handle discrete problems.

2.1. Particle Swarm Optimization

The Particle Swarm Optimization metaheuristics1 (PSO) was motivated by the coordinate movement of fish schools and bird flocks (Kennedy and Eberhart, 2001). A potential solution to the problem being solved is represented by a particle, and the PSO is a swarm of particles. Particles "flow" through hyper-dimensional search space of the problem, and changes to the position of the particles within the search space are based on the socio-cognitive tendency of individuals to emulate the success of other individuals. Each individual of a population (in this case, particles) has its own life experience and is able to evaluate the quality of its experience. As social individuals they also have knowledge about how well their neighbors have behaved. These two kind of information corresponds to the cognitive component (individual learning) and social component (cultural transmission), respectively. Hence, an individual decision is taken considering both the cognitive and the social components, thus leading the population (the swarm) to an emergent behavior.

2.2. Roach Infestation Optimization

The Roach Infestation Optimization (RIO) was introduced by Havens et al. (2008), who applied to benchmark functions and achieved competitive results, when compared to a standard PSO. Actually, RIO has some elements that resemble the traditional PSO algorithm. In the RIO algorithm, cockroaches agents are defined using three simple behaviors:

1. Cockroaches search for the darkest location in the search space and the fitness value is directly proportional to the level of darkness (find darkness phase).
2. Cockroaches socialize with nearby cockroaches (find friend phase).
3. Cockroaches periodically become hungry and leave the friendship to search for food (find food phase).

2.3. Cuckoo Search Algorithm

Cuckoo Search Algorithm (CSA) (Yang and Deb, 2009) is based on the brood parasitism of some cuckoo species. The algorithm uses the Levy flights rather than simple random walk. The CSA uses the following main basic rules:

1. Each cuckoo lays one egg at a time, and dumps its egg in a randomly chosen nest;

2. The best nests with high-quality eggs will continue to the next generation;
3. The number of available host nests is fixed, and the egg laid by a cuckoo is discoved by the host bird with a probability $p_a \in [0,1]$. In this case, the host bird can either get rid of the egg, or simply abandon the nest and build a new nest.

2.4. Firefly Algorithm

The Firefly Algorithm (FA) was proposed by (Yang, 2008) and uses three main basic rules:

1. A firefly will be attracted by other fireflies regardless their sex;
2. Attractiveness is proportional to their brightness and decreases as the distance among them increases;
3. The landscape of the objective function determines the brightness of a firefly.

2.5. Gravitational Search Algorithm

The Gravitational Search Algorithm (GSA) was created based on the law of gravity and the notion of mass interactions (Rashedi et al., 2009). The GSA algorithm uses the theory of Newtonian physics and its searcher agents are the collection of masses. In GSA, there is an isolated system of masses, using the gravitational force, every mass in the system can detect the situation of other masses. The gravitational force is therefore a way of transferring information between different masses. The GSA agents are considered as objects and their performance is measured by their masses. All these objects attract each other by a gravity force, and this force causes a movement of all objects globally towards the objects with heavier masses. The heavy masses correspond to good solutions of the problem. The position of the agent corresponds to a solution of the problem, and its mass is determined using a fitness function.

2.6. Bat Algorithm

The Bat Algorithm (BA) was first presented in (Yang, 2010). The basic idea behind the Bat Algorithm is that a population of bats (possible solutions) use echolocation to sense distance and fly randomly through a search space updating their positions and velocities. The bats' flight aims at finding food/prey (best solutions). A loudness decay factor acts in a similar role as the cooling schedule in the traditional simulated annealing optimization method, and a pulse increase factor regulates the pulse frequency. As the loudness usually decrease once a bat has found its prey/solution (in order to do not loss the prey), the rate of pulse emission increases in order to raise the attack accuracy.

2.7. Glow-worm Swarm Optimization Algorithm

The Glow-worm Swarm Optimization (GSO) algorithm was first presented by (Krishnanand and Ghose, 2005) as an application to collective robotics. In this algorithm, each glow-worm uses a probabilistic mechanism to select a neighbor that has a luciferin value associated with him and moves towards it. Glow-worms are attracted to neighbors that glow brighter (that is, glow-worms that have more luciferin). The movements are based only on local information and selective neighbor interactions. This enables the swarm to divide into disjoint subgroups that can converge to multiple optima of a given multimodal function.

2.8. Artificial Fish School Algorithm

In water areas, a fish can always find food at a place where there are plenty of food by following other fishes, hence generally the more the food, the more the fish. Following this rule, Artificial Fish School Algorithm (AFSA) builds some artificial fish (AF), which search an optimal solution in solution space (the environment in which AF live) by imitating fish swarm behavior. Three basic behaviors of AF are defined as follows (Cai, 2010):

1. Prey: The fish perceives the concentration of food in water to determine the movement by vision or

sense and then chooses the tendency.
2. Swarm: The fish will assemble in groups naturally during the moving process, which is a kind of living habits in order to guarantee the existence of the colony and avoid dangers.
3. Follow: In the moving process of the fish swarm, when a single fish or several fishes find food, the neighborhood partners will trail and reach the food quickly.

2.9. Bacterial Evolutionary Algorithm

The Bacterial Evolutionary Algorithm (BEA) is inspired by bacteria's behavior. Bacteria can transfer DNA to recipient cells through mating and this process is called transduction. By transduction, it is possible to spread the features of a single bacterium to the rest of the population. This genetic recombination mechanism characterizes a microbial evolution process. Male cells directly transfer strands of genes to female cells. After that, those female cells acquire characteristics of male cells and transform themselves into male cells, thus spreading to the entire bacteria population. Genes can be transferred from a single bacterium (host cell) to others (recipient cells) and eventually this would lead to an increase in the evolution speed of the entire population (Nawa and Furuhashi, 1999). The flow of the BEA is described as follows:

1. Generation of the initial population: chromosomes are randomly created and evaluated;
2. Bacterial Mutation: The bacterial mutation is applied to each chromosome, one by one;
3. Gene Transfer Operation: The gene transfer operation occurs between the chromosomes.

2.10. Bee Algorithm

The Bee Algorithm[1] (BA_1) was first introduced by (Pham et al., 2005) and applied to a benchmark of mathematical functions. In this algorithm, a bee is a *d*-dimensional vector containing the problem variables (a solution).Moreover, a solution represents a visited site (i.e., food source) and has a fitness value assigned to it. The algorithm balances exploration and exploitation by using scout bees that randomly search for new sites and use recruitment for neighborhood search in sites with higher fitness, respectively. Bees that have the highest fitness are chosen as "selected bees" and sites visited by them (elite sites) are chosen for neighborhood search. The algorithm conducts searches in the neighborhood of the selected sites, assigning more bees to search near to the best sites (recruitment).

2.11. Artificial Bee Colony Algorithm

The Artificial Bee Colony Algorithm (ABC) was first proposed by (Karaboga, 2005) for solving multi-dimensional and multi-modal optimization problems. The bees' aim is to discover places of food sources (regions in the search space) with high amount of nectar (good fitness). There are three types of bees: the scout bees that randomly fly in the search space without guidance; the employed bees that exploit the neighborhood of their locations selecting a random solution to be perturbed; and the onlooker bees that use the population fitness to select probabilistically a guiding solution to exploit its neighborhood. If the nectar amount of a new source is higher than the previous one in their memory, they update the new position and forget the previous one (greedy selection). If a solution is not improved by a predetermined number of trials, then the food source is abandoned by the corresponding employed bee and it becomes a scout bee.

2.12. Bee Colony Optimization

The Bee Colony Optimization (BCO) was proposed by (Lucic and Teodorovic, 2002) and, similarly to ABC, it imitates the bees behavior in nature when looking for a food, by simulating the foraging behavior. The BCO is based on the constructive concept. It was designed as a method which builds solutions from the scratch within execution steps, unlike the ABC local search which perform iterative improvements of the current best solution. There are two alternating phases of the BCO (forward pass and

1The subindex 1 is introduced here to differentiate the Bee Algorithm from the Bat Algorithm

backward pass) constituting a single step in the BCO algorithm. In each forward pass, every bee agent visits the solution components, creates a partial solution, and after that returns to the hive. Having obtained new partial solutions, the bees meet in the hive and start the backward pass. In the backward pass, all bee agents share information about the quality of their partial solutions. Having all solutions evaluated, each bee decides with a certain probability whether it will stay loyal to its solution or not.

2.13. Marriage in Honey-bees Optimization Algorithm

The Marriage in Honey-bees Optimization Algorithm (MBO) was presented by Abbass (2001b). The main idea concerning the algorithm based on bee mating behavior is that the queen is considered the best solution to an optimization problem and, during the mating flight, it selects drones probabilistically for reproduction so as to form the spermatheca. The spermatheca is, then, a pool of selected solutions. New broods are created by crossovering the genotypes of drones and the queen. Natural selection takes place by replacing weaker queens by fitter broods.

3. Main Concerns to Handle Discrete Problems

In most swarm intelligence algorithms the possible solution is encoded as a set of real variables, which represents, for instance, the location of a particle (in PSO) or a source food (in ABC). In these examples, updates in a given vector position (dimension) can be done independently of the remaining ones. The updated values remains within the predefined range for that dimension, the candidate solution is valid (for the sake of simplification, constraints are not considered here). However, for combinatorial problems, such representation is not suitable, provided that a solution is a permutation of integer values. Therefore, updates of a vector that represents a solution for a combinatorial problem must preserves the validity of the permutation. Consequently, to apply the above-mentioned algorithms to discrete optimization problems it is necessary to adapt the encoding to discrete dimensions.

We characterize the codification of candidate solutions in three encoding schemes. The first encoding uses a binary codification (BC) for candidate solutions. The second encoding uses an integer codification (IC) for candidate solutions. Swarm algorithms that employ BC or IC are adapted to discrete domain since its beginning (i.e., initial swarm). The third encoding deals with real values but uses transformation methods to handle discrete domains. The real solution vector can be transformed into a binary codification (real-to-binary: RTB) or it can be transformed into an integer codification (real-to-integer: RTI) where RTI represents a combination of integer values. Swarm algorithms that employ RTB or RTI require these transformations at each iteration loop.

3.1. Discretization Methods

In discrete problems, such as combinatorial, binary and categorical, it is necessary to reduce the number of possible states to feasible solutions. This is done by discretization of the continuous space by transforming the values into a limited number of possible states. There are several discretization methods, and the main are presented below.

3.1.1. Sigmoid function

The sigmoid function (SF) can be used to transform a continuous space value into a binary one. Such discretization method is very popular (Palit et al., 2011; Banati and Bajaj, 2011), and the transformation is applied to each dimension of the solution vector, as shown in Equation 1, thus forcing each element to be a binary digit:

$$x_{ij} = \begin{cases} 1, & \text{if } rand() \leq \frac{1}{1+\exp(-x_{ij})} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

with $i = 1,\ldots,N, j = 1,\ldots,D$, where N is the population size, D is the dimension size and *rand*() is a random

number drawn uniformly from [0,1].

3.1.2. Random-key

The random-key (RK) encoding scheme can be used to transform a position from a continuous space into an integer/combinatorial space (Chen et al., 2011; Li et al., 2010). To decode the position, the nodes are visited in ascending order for each dimension. For example, the continuous solution vector $\vec{x}_i$ = (0.90,0.35,0.03,0.21,0.17) can be decoded as $\vec{x}_i$ = (5,4,1,3,2).

3.1.3. Smallest Position Value

The Smallest Position Value (SPV) method maps the positions of the solution vector by placing the index of the lowest valued component as the first item on a permutated solution, the next lowest as the second and so on. This method creates an integer vector solution by indexing the position of all the particles (Verma and Kumar, 2012; Yousif et al., 2011)

3.1.4. Modified Position Equation

The Modified Position Equation (MPE) method has been used only in the PSO algorithm (Pan et al., 2008; Tasgetiren et al., 2007). As mentioned before, the standard PSO updates the particle position using three choices: follow its own position (X_i^t), go towards its personal best position (P_i^t) or go towards the best particle position of the swarm (G_i^t). After that, the particle position on iteration t can be updated following Equation 2:

$$X_i^t = c_2 \otimes F_3(c_1 \otimes F_2(w \otimes F_1(X_i^{t-1}), P_i^{t-1}), G_i^{t-1}) \quad (2)$$

where $\lambda_i^t = w \otimes F_1(X_i^{t-1})$ is the velocity of the particle; F_1 is the mutation operator with probability w. A uniform random number r in the range [0,1] is generated and, if it is less than w, then the mutation operator is applied to generate a perturbed permutation of the particle by $\lambda_i^t = F_1(X_i^{t-1})$; otherwise current permutation is kept as $\lambda_i^t = X_i^{t-1}$. In this equation, $\delta_i^t = c_1 \otimes F_2(\lambda_i^t, P_i^{t-1})$is the cognitive part of the particle; F_2 is the crossover operator with probability c_1. Note that λ_i^t and P_i^{t-1} will be the first and second parents for the crossover operator. It results either in $\delta_i^t = F_2(\lambda_i^t, P_i^{t-1})$ or in $\delta_i^t = \lambda_i^t$ depending on the choice of a uniform random number. The third component of Equation 2, $X_i^t = c_2 \otimes F_3(\delta_i^t, G_i^t)$, is the social part of the particle, where F_3 is the crossover operator with probability c_2. Note that δ_i^t and G_i^{t-1} will be the first and second parents for the crossover operator. It results either in $X_i^t = F_3(\delta_i^t, G_i^{t-1})$ or in $X_i^t = \delta_i^t$ depending on the choice of a uniform random number.

3.1.5. Great Value Priority

The Great Value Priority (GVP) method is used to encode a continuous space into a binary one (Congying et al., 2011). First, the position of the solution vector $\vec{x}_i$ with the largest element is selected, where $i = 1,\ldots,N$ and N is the population size. This position is set on the first position of a new vector named as permutation vector $\vec{p}$. Next, the position of the second largest element of $\vec{x}_i$ is selected and placed in the next position of $\vec{p}$. This procedure is repeated successively for all dimensions of $\vec{x}_i$ and, once $\vec{p}$ is fulfilled, Equation 3 is applied to transform it into binary, where $j = 1,\ldots,D$ and D is the dimension size.

$$\vec{x}_{ij} = \begin{cases} 1, & \text{if } \vec{p}_j > \vec{p}_{j+1} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

3.1.6. Nearest integer

In this method, a real value is converted to the nearest integer (NI) by rounding or truncating up or down (Burnwal and Deb, 2012).

4. Applications to Discrete Problems

For the algorithms mentioned in Section 2, a literature search was done so as to find discrete applications in different domains. Although this search was not exhaustive, it covers the most relevant applications, and emphasizes the applicability of those algorithms to discrete problems.

4.1. Particle Swarm Optimization

Kennedy and Eberhart (1997) developed the first PSO algorithm for combinatorial optimization problems, where particles were encoded as binary sequences. In the binary version, trajectories are changes in the probability that a coordinate will take a zero or one value. Also, the velocity of the particle may be described by the number of bits changed per iteration, or the Hamming distance between the particle at time t and at $t + l$.

Wan and Nolle (2009); Deep and Bansal (2008); Shen et al. (2012); Hembecker et al. (2007) used the SF method to calculate the trajectory in the binary search space to solve the Knapsack Problem (KP). For the same problem, Lopes and Coelho (2005) used a PSO hybridized with a guided local search and some concepts drawn from Genetic Algorithms (GA). In (Li and Li, 2009) a binary PSO is proposed to solve the KP using a mutation operator.

A PSO was used to solve the flowshop scheduling problem by Ucar and Tasgetiren (2006). First, a population of particles with continuous values is randomly constructed and then transformed into discrete values by applying the SPV method. The same method was used to solve the single machine total weighted tardiness problem by Tasgetiren et al. (2004a,b), where the SPV rule is applied to convert the position vector to a job permutation. Verma and Kumar (2012) also used PSO with SPV to solve a combinatorial optimization problem, the DNA sequence assembly. Lin et al. (2010) used a continuous PSO combined with the RK transformation to solve the job-shop scheduling problem. PSO was hybridized with a local search based on simulated annealing technique.

Congying et al. (2011) used the PSO in a typical combinatorial optimization problem, the Quadratic Assignment Problem (QAP), and the GVP method was used to transform the continuous space to discrete.

Rosendo and Pozo (2010) presented a hybrid PSO algorithm to solve the Traveling Salesman Problem (TSP). This work maintains the main PSO concept for updating the velocity of the particle using path relinking (Glover and Laguna, 1997) and the Lin-Kernighan algorithm (LK) (Lin et al., 1973) to improve local search.

Labed et al. (2011) proposed MHPSO, a hybrid PSO to solve the Multidimensional KP. As in Lopes and Coelho (2005), it combines PSO with the crossover operator of GA. The crossover operator is applied in both, the best position of each particle (*pbest*) and its current position, as well as in the best position obtained by the swarm (*gbest*) and its current position.

To the best of our knowledge, Pan et al. (2008) reported for the first time in the literature a Discrete Particle Swarm Optimization (DPSO) algorithm to solve the no-wait flowshop scheduling problem. The particles are represented as discrete job permutations and a new position update method is developed based on the discrete domain that consists of three operators. A new crossover operator was also

introduced, and it is able to produce a pair of different permutations even from two identical parents. A block of jobs from the first parent is determined by two-cut points randomly, and this block is moved to the right or left of the solution vector. Then, the offspring permutation is filled out with the remaining jobs from the second parent. The DPSO algorithm is also hybridized with the Variable Neighbourhood (VND) local search by Mladenovic and Hansen (1997) to improve the performance. Tasgetiren et al. (2007) also used DPSO with the MPE method to solve the generalized TSP.

4.2. Roach Infestation Optimization

Hendrawan and Murase (2011) proposed a discrete RIO to solve a feature selection problem using a neural network as a predictive tool. A binary encoding vector was used for each roach, and each dimension represents a feature (value 0 or 1). A 0 indicates that the corresponding feature is not selected, while 1 means the opposite. To update the position of the roach, the cockroach's velocity is updated with a mutation operator and then the cognition and social parts of the cockroach are updated using a two-points crossover.

4.3. Cuckoo Search Algorithm

Burnwal and Deb (2012) demonstrated for the first time a Cuckoo Search algorithm (CSA) to solve a scheduling problem for a Flexible Manufacturing System (FMS). They modified the Levy flights from continuous to discrete values using the nearest integer. For the same type of problem, Kumar and Chakarverty (2011) encoded a solution in the CSA in the form of an integer vector.

Gherboudj et al. (2012) proposed a discrete binary CSA to solve the KP. To get binary solutions the SF was used and Layeb (2011) presented a quantum inspired cuckoo search algorithm (QICSA) for the same KP but, instead, a binary encoding was used.

4.4. Firefly Algorithm

The first Firefly Algorithm (FA) for combinatorial optimization was put forward in Sayadi et al. (2010) for makespan minimization in a permutation flow shop scheduling problem. Also, Falcon et al. (2011) used the same ideas for a fault diagnosis problem. In both cases they used a binary approach: for updating the firefly's position, SF was applied to each component of the real-valued vector. The SF method was used to transform the values from real to binary by Palit et al. (2011), for the KP, and by Banati and Bajaj (2011), to solve the binary problem of feature selection.

Yousif et al. (2011) introduced a method based on FA for scheduling jobs on grid computing. In this case, the SPV approach was used for updating the positions of the fireflies from continuous position values to discrete permutations.

Fister et al. (2012) used the FA to solve a combinatorial optimization problem, the graph 3-coloring. The RK method was used to convert the real values to discrete.

4.5. Bee Algorithm

The Bee Algorithm (BA_1) was first applied to continuous optimization functions and, later for scheduling jobs (Pham et al., 2007a) and binary data clustering (Pham et al., 2007b). In these papers, the authors used discrete encoding, ensuring that possible solutions would be valid. The BA_1 also inspired Bahamish et al. (2008) to create a discrete encoding for the Protein Tertiary Structure Prediction (PTSP), a difficult combinatorial optimization problem.

A well-known discrete problem is the Generalized Assignment Problem (GAP), approached by Ozbakir

et al. (2010) using a modified BA. In the original BA_1 scout bees continue searching in the solution space until the algorithm is terminated. However, in the modified BA_1 an adaptive penalty coefficient mechanism is employed.

A Binary BA_1 (BBA) was proposed by Xu et al. (2010) focusing on a two-level Distribution Optimization Problem (DOP), which expressed the agents (bees) as two binary matrices, representing how to assign each bee to its course or mission.

4.6. Artificial Bee Colony

The applicability of the ABC algorithm the GAP (Baykasoğlu et al., 2007) demonstrates the capability to adapt this algorithm to discrete problems, using the RK method. The ABC was also applied to other discrete problems such as the Job Shop Scheduling Problem (JSSP), TSP, multidimensional KP. For these applications, many variants and hybridisms of the ABC were proposed.

An adaptation of the original ABC method was presented by Banharnsakun et al. (2012) for the JSSP, by changing the behavior of the onlooker bees and and using the SF method to transform the continuous values into binary. The proposed new method, called Best-so-far ABC, uses all the onlooker bees with the information from all employed bees to make a common decision on a new candidate food source. Because of that, the onlookers can compare information from all candidate sources and they are able to select the best-so-far position. As a result, this modification makes the algorithm converge more quickly since that solution will be biased towards the best solution found so far.

A Combinatorial ABC (CABC) with discrete encoding was proposed by Karaboga and Gorkemli (2011) for the TSP. In that work a new mutation operator from GA was used, and it was adapted to the neighborhood search mechanism of employed and onlooker bees. The hybridism between GA and ABC was also approached by Singh et al. (2011), which used ABC techniques to improve the global search by adding employed and onlooker bees on the search process, this method was called Real Genetic Bee Colony Algorithm (RGBCA). A similar hybridism, between ABC and Greedy Subtour Crossover (GSX), was used by Banharnsakun et al. (2010) for the same problem. In this hybrid method, the exploitation process of the ABC algorithm was improved by the GSX, so as to update employed bees' old food sources based on their neighboring food sources, assisting the ABC to generate feasible solutions all the time.

The Knapsack Problem (KP) is another classical discrete problem that has been studied using the ABC algorithm. Pulikanti and Singh (2009) used a modified and binary version of ABC for this problem. In the original ABC algorithm, the neighborhood of the best found food sources are searched in order to achieve even better food sources. In the novel ABC algorithm, a hybrid probabilistic mutation scheme was performed for searching the neighborhood of food sources. Here the SF method was used again to convert real values into binary.

Hybrid methods have been used to improve the original ABC algorithm, generally associated to a neighborhood search mechanism to improve global searches. Pacurib et al. (2009) uses this improved search, adding a search iteration operator based on the fixed point theorem of contractive mapping in Banach spaces (discrete vector space) with ABC algorithm to solve Sudoku puzzles.

Another hybrid discrete ABC (DABC) was proposed by Marinakis et al. (2009), using a two-phases algorithm that combines the DABC and the Greedy Randomized Adaptive Search Procedure (GRASP). This DABC-GRASP algorithm was created for the solution of clustering problems and the two-phases consist in the computation of the activated features by the DABC and the computation of the fitness of each food source by the clustering algorithm.

4.7. Bee Colony Optimization

The BCO proposed by Teodorovic and Dell'Orco (2005) was applied to the Ride-matching problem and Wavelength Assignment (RWA) using a discrete encoding. Later, Banerjee et al. (2008) created a hybrid BCO for modeling process and supply chain scheduling, in this case, the the SF method was used to transform the continuous space into binary. Davidovic et al. (2012) also used the BCO algorithm for scheduling independent tasks on homogeneous processors, their approach brings a modified BCO which allows the bees a freedom to generate various solutions leading to a diversification of the search process. Wong et al. (2008a) also studied the BCO and applied it to the TSP.

Chong et al. (2006) described an BCO algorithm based on foraging and waggle dance, using the dance duration as determining factor for selecting a new path. This approach was applied to the JSSP with discrete encoding and a neighborhood structure to search for feasible solutions and improve prior solutions (Wong et al., 2008b). A set of priority dispatching rules were used to create the initial solutions.

4.8. Marriage in Honey-bees Optimization Algorithm

The MBO algorithm was applied to the propositional satisfiability problems, known as SAT problems (Abbass, 2001b). MBO was applied to 50 variables of the SAT problems with 215 different constraints, presenting good results for this combinatorial optimization problem with a binary encoding. Later, the algorithm was improved in Abbass (2001a) by using a single queen with multiple workers in the colony, and also applied to the 3-SAT problem where each constraint contains exactly three variables.

A conventional annealing approach was used by Teo and Abbass (2001) on the mating-flight process to balance search exploration and exploitation, and this modified MBO was applied to SAT problems. This algorithm was also applied in data-mining (Benatchba et al., 2005) and scheduling problems (Koudil et al., 2007) with the SF method to convert the continuous space into binary.

The SF method was also used by Marinakis et al. (2008) in a MBO hybridized with GRASP for clustering problems with N objects into K clusters. Chenguang Yang (2007) combined the MBO with the Nelder-Mead method with the objective of improving its performance, and thus creating the NMFMBO algorithm to be applied to the TSP.

4.9. Other Swarm Intelligence Algorithms

The Gravitational Search Algorithm (GSA) was applied by Li et al. (2010) with the RK discretization method transforming the continuous solutions into discrete ones. Chen et al. (2011) also applied the GSA to solve the TSP with RK, but integrating the simulated annealing technique (SA) into the algorithm for accomplishing local search. A binary implementation of GSA was proposed by Papa et al. (2011), together with the SF method, and used for a feature selection problem.

Nakamura et al. (2012) proposed a binary version of the Bat Algorithm (BA) to solve the hypercube problem, where each bat is a set of binary coordinates. The equation to update the position of the original BA is replaced by the SF method of discretization.

The GSO was applied by Deng-xu et al. (2011) to solve the multi-constrained QoS multicast routing problem using a discrete vector, this problem is a direct application of the well-known TSP.

Xiangyang et al. (2011) proposed the Artificial Fish School Algorithm (AFSA) to solve KP using the RK method. He et al. (2009) proposed the AFSA mapped into the integer space directly, using a method that ensures that the candidate solution stays in integer space throughout the optimization process.

Balazs et al. (2012) proposed a Bacterial Evolutionary Algorithm (BEA) to solve a permutation flow shop problem with a real valued vector (arrays). The adaptation of real valued vectors to permutations is done

by using the RK method. Inoue et al. (2000); Miwa et al. (2002) proposed a binary encoding for the BEA to solve the nurse scheduling problem using mutation and crossover similar to those of GA. Feng and Xiao (2011) applied the same method for a route planning problem.

5. Discussion

The algorithms presented in Section 2 were originally devised for dealing with continuous search spaces. However, for discrete search spaces, which is the case of binary and combinatorial problems, the key issue is how to adapt the before mentioned algorithms

In this survey we selected sixty four (63) papers for analysis. Figure 1 shows the distribution of the swarm algorithms covered here. It is clearly observed that PSO was the most frequently found algorithm, representing a quarter of all papers analyzed. The ABC algorithm was the second one, representing 13% of the total.

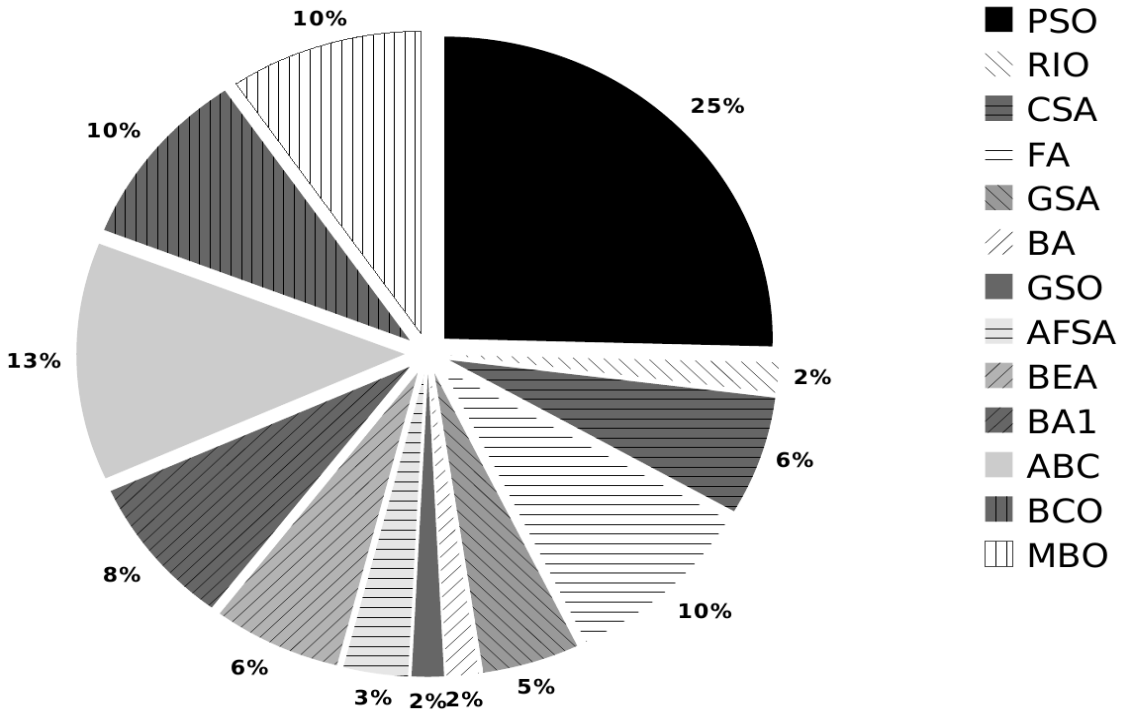

Figure 1: Algorithms Distribution.

Figure 2 summarizes the distribution of the discretization methods (Section 3) employed by the algorithms. The most common used method is the SF. 62% of the papers used it to transform continuous values into discrete ones. It is worth to mention that SF is also extensively applied in several other areas such as artificial neural networks, biomathematics, chemistry and statistics.

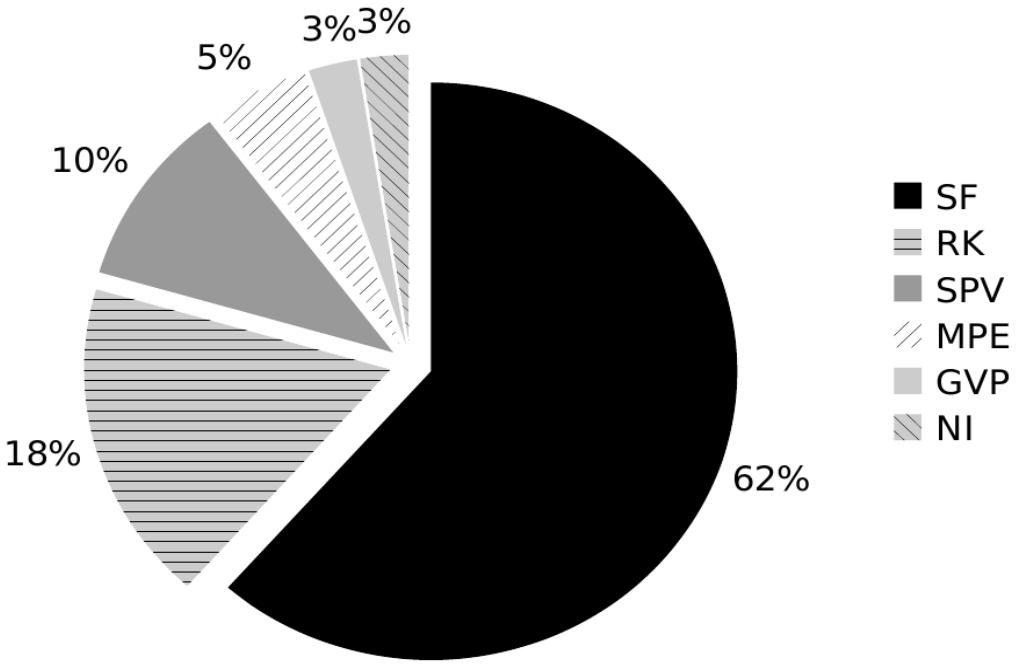

Figure 2: Discretization methods.

The RK method was the second most frequent, found in 18% of the papers. Since this method is exclusive to transform into integer, not binary, when binary values were necessary a combination of the RK method with the GVP was used by Congying et al. (2011). In this combined process, the first stage consisted in transforming the continuous values into discrete using the RK and a second stage using GVP to convert the discrete values in binary ones. The SPV was also used in 10% of the papers to transform continuous into discrete. The MPE method was used in 5% of the papers. This method was combined with a discrete encoding to handle scheduling and TSP problems. The MPE method was used only in PSO algorithms, but it could be adapted to any other algorithm that uses the same main procedures of PSO, that is, the update of speed and position. Finally, only 3% of the papers used the NI for this same sort of transformation.

Figure 3 summarizes how many times each discretization method was used by each algorithm in the papers surveyed. Notice that two of them, RIO and GSO, did not use any discretization method. This is due to the straight encoding schemes (integer or binary) that do not require further discretization. In two (2) papers there was no enough information for identifying the discretization method used, so they were not counted in Figure 3.

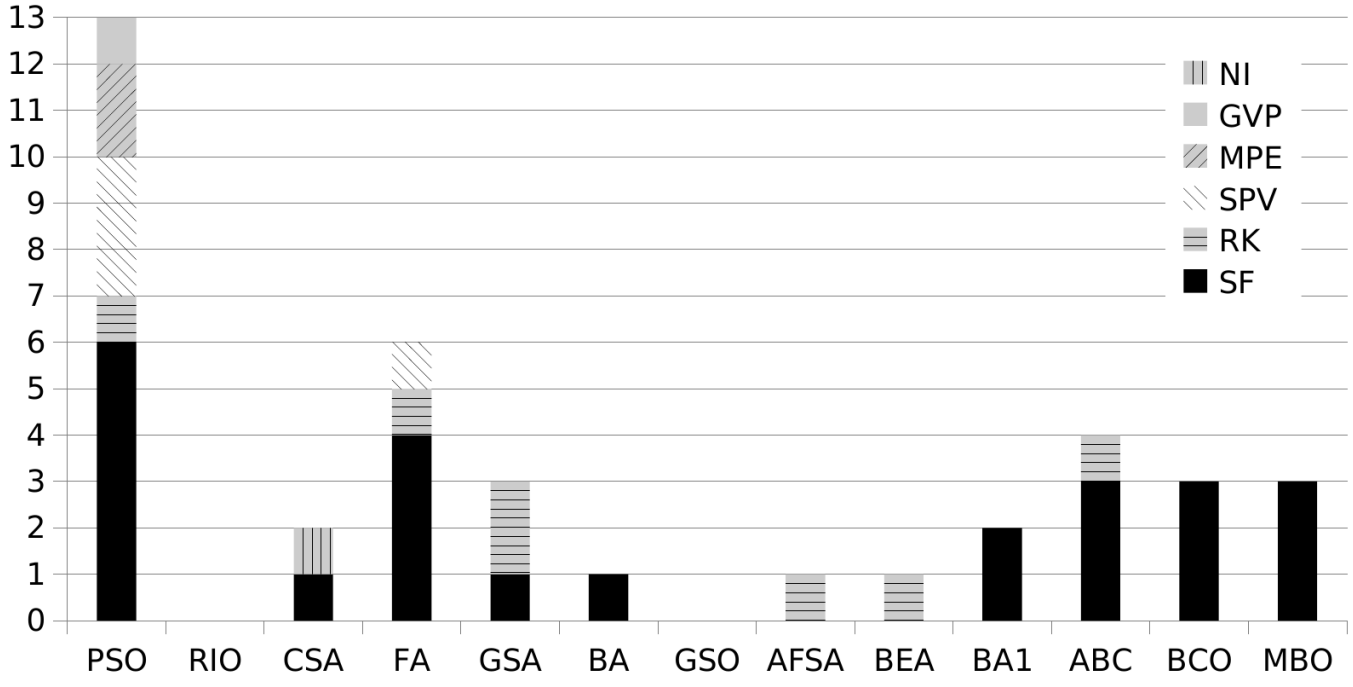

Figure 3: Discretization methods used in the algorithms.

Figure 4 lists the encoding schemes (BC, IC, RTB or RTI – see Section 3). 19% of the papers preferred to use the encodings directly as binary and 25% as integer. In this approach, the solution vector is discrete and the algorithm code needs to be modified accordingly to work with these values. For this purpose, several authors created a discrete version of the continuous algorithm, e.g., the Discrete Particle Swarm Optimization (DPSO), the Combinatorial ABC (CABC) and the Discrete Artificial Bee Colony Optimization (DABC). The other 55% of papers preferred to use the original continuous encoding then transforming them the real-valued vector into integer or binary, by using one of the methods presented on Section 3.

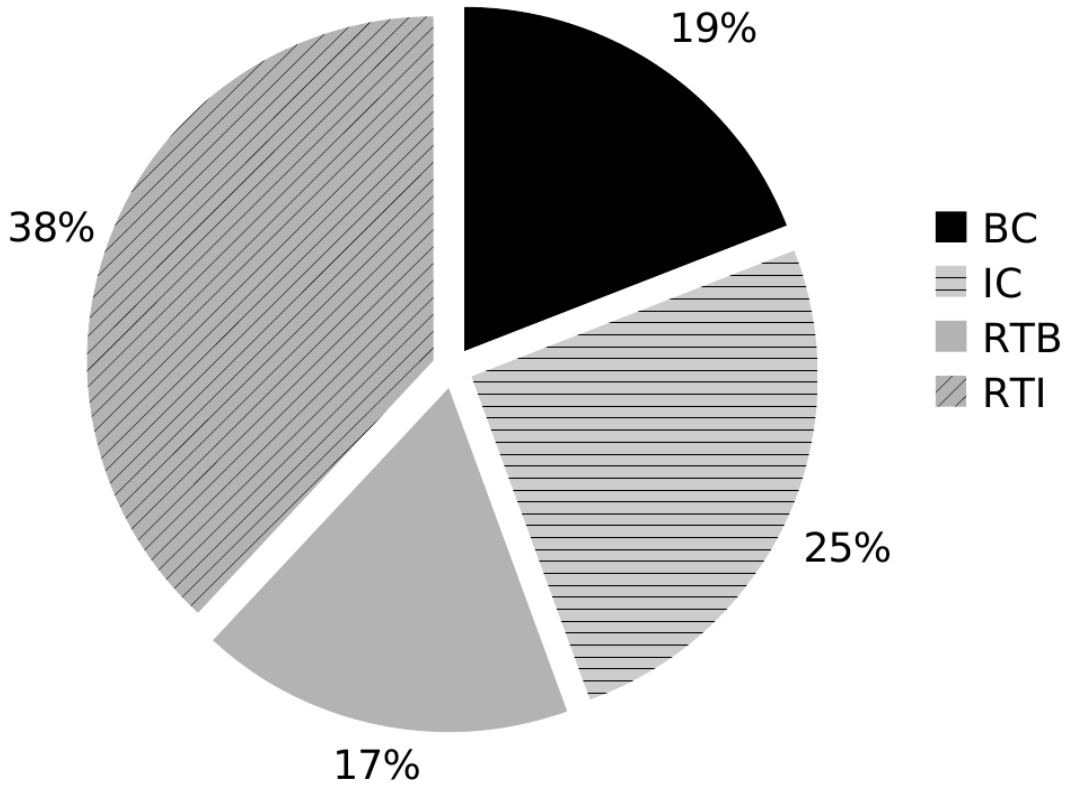

Figure 4: Distribution of encoding schemes.

Figure 5 is a more detailed view of the previous plot. It summarizes which encoding scheme was used in each algorithm and how many times these schemes were used.

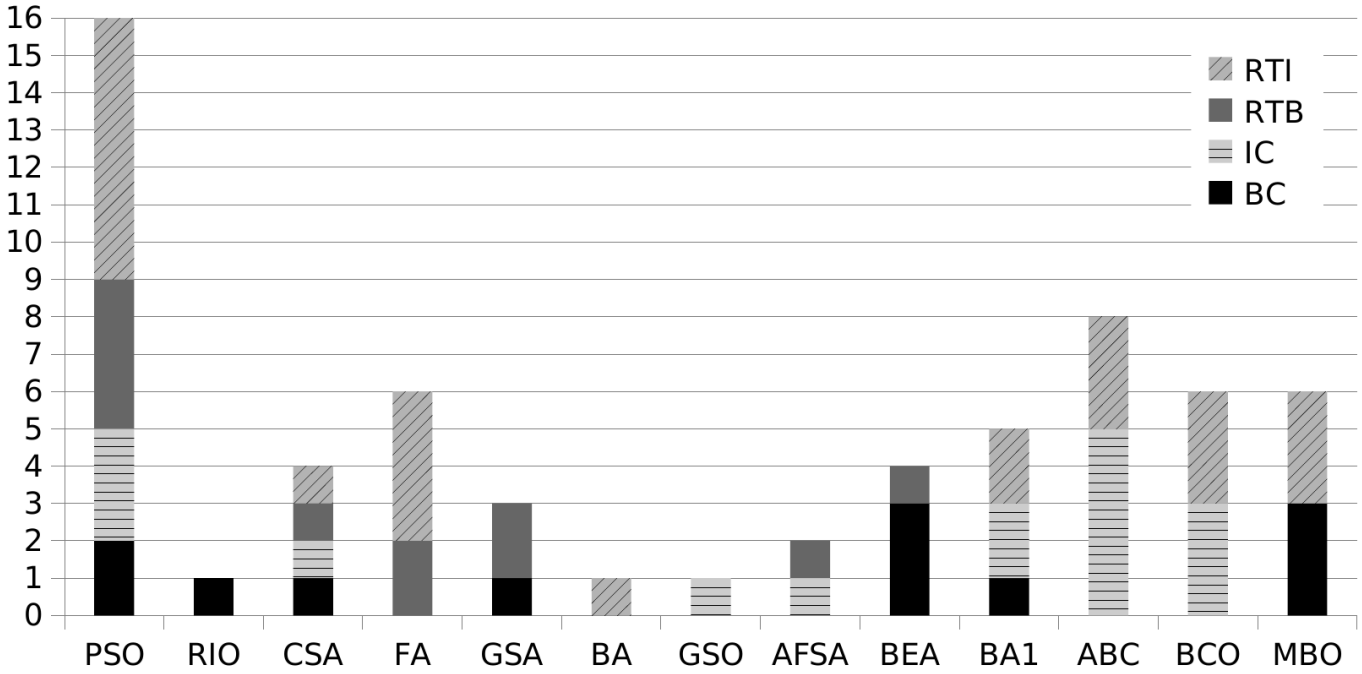

Figure 5: Encoding schemes related of swarm algorithms.

Finally, regarding the application, Figure 6 shows the main classes of problems to which the swarm optimization algorithms were applied. Scheduling problems are the most frequently found, close followed by the knapsack problem.

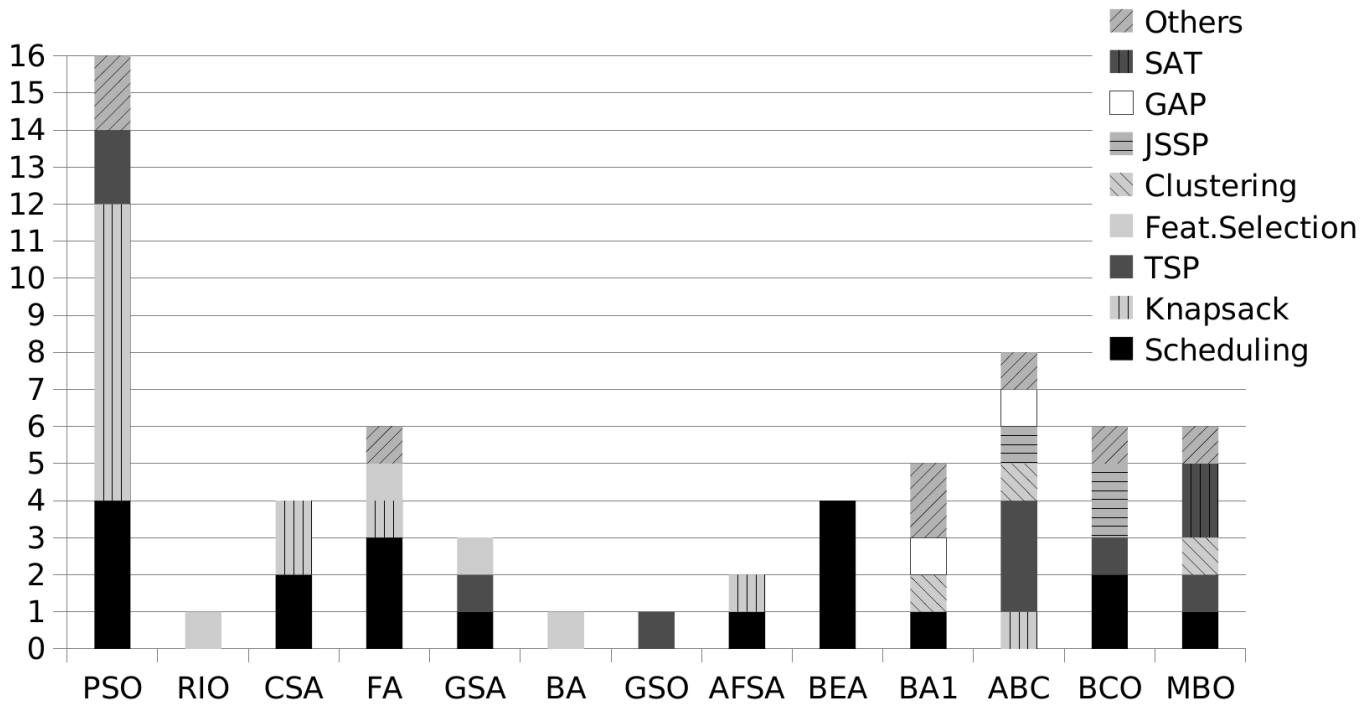

Figure 6: Applications of the swarm algorithms.

6. Concluding Remarks and Future Research

Real-world discrete and combinatorial problems are usually challenging and require computationally intensive algorithms. Swarm intelligence algorithms have been an interesting alternative for providing satisfactory solutions. The individual characteristics of the continuous swarm algorithms are very useful for these applications. However, to apply them to a discrete problems requires adaptation, hybridization or modification the original algorithms to handle those sort of problems.

The methods used to achieve this purpose are the most diverse. This survey aims to classify the most common discretization methods and encodings strategies to guide future authors to select the best method and strategy when working with continuous swarm algorithms for discrete problems. The SF and the RK discretization methods appeared most times in the literature review. Both methods seek the transformation of the continuous values handled by the algorithms into discrete ones, requiring an adaptation to deal with possible solutions in a discrete environment.

The results achieved by each paper presented in this survey demonstrate how continuous algorithms can be useful in discrete problems. Particularly, the use of the PSO for classical combinatorial problems, such as KP, Scheduling and TSP by several authors demonstrate a trend due the large number of papers addressing these problems. For the Scheduling Problem, the FA and BEA also appeared several times, creating a trail for future authors that intend to work with this combinatorial problem.

It is worth to note that for several recent swarm algorithms presented literature, only a few of them were applied to discrete problems. This seems to be an interesting research opportunity, either developing new methodological approaches or applying to specific discrete/combinatorial problems.

Future work points to a more extensive research on different discretization methods and encoding schemes, focusing on expanding the range of possibilities on using continuous algorithms to discrete problems. Therefore, authors will have in the future more alternatives to combine the desired continuous algorithm with combinatorial and binary problems.

References

Abbass, H., 2001a. A monogenous MBO approach to satisfiability. In: International Conference on Computational Intelligence for Modelling, Control and Automation.

Abbass, H. A., 2001b. MBO: marriage in honey bees optimization a haplometrosis polygynous swarming approach. In: IEEE Congress on Evolutionary Computation. Vol. 1. pp. 207–214.

Bahamish, H. A. A., Abdullah, R., Salam, R. A., 2008. Protein conformational search using bees algorithm. In: Second Asia International Conference on Modelling & Simulation. IEEE Computer Society, Washington, DC, USA, pp. 911–916.

Balazs, K., Horvath, Z., Koczy, L., 2012. Hybrid bacterial iterated greedy heuristics for the permutation flow shop problem. In: IEEE Congress on Evolutionary Computation. pp. 1–8.

Banati, H., Bajaj, M., 2011. Firefly based feature selection approach.

Banerjee, S., Dangayach, G. S., Mukherjee, S. K., Mohanti, P. K., 2008. Modelling process and supply chain scheduling using hybrid meta-heuristics. In: Metaheuristics for scheduling in industrial and manufacturing applications. Vol. 128 of Studies in Computational Intelligence. Springer, Berlin, Germany, pp. 277–300.

Banharnsakun, A., Achalakul, T., Sirinaovakul, B., 2010. ABC-GSX: A hybrid method for solving the traveling salesman problem. In: 2nd World Congress on Nature and Biologically Inspired Computing. pp. 7–12.

Banharnsakun, A., Sirinaovakul, B., Achalakul, T., 2012. Job shop scheduling with the best-so-far ABC. Engineering Applications of Artificial Intelligence 25 (3), 583–593.

Baykasoğlu, A., Ozbakir, L., Tapkan, P., 2007. Artificial bee colony algorithm and its application to generalized assignment problem. In: Chan, F. T. S., Tiwari, M. K. (Eds.), Swarm Intelligence: Focus on Ant and Particle Swarm Optimization. I-tech Education and Publishing, pp. 532–564.

Benatchba, K., Admane, L., Koudil, M., 2005. Using bees to solve a data-mining problem expressed as a max-sat one. In: Artificial Intelligence and Knowledge Engineering Applications: a bioinspired approach. Vol. 3562 of Lecture Notes in Computer Science. Springer-Verlag, Heidelberg, pp. 75–86.

Burnwal, S., Deb, S., 2012. Scheduling optimization of flexible manufacturing system using cuckoo search-based approach. The International Journal of Advanced Manufacturing Technology, 1–9.

Cai, Y., 2010. Artificial fish school algorithm applied in a combinatorial optimization problem. International Journal of Intelligent Systems and Applications 1, 37–43.

Chen, H., Li, S., Tang, Z., 2011. Hybrid gravitational search algorithm with random-key encoding scheme combined with simulated annealing. International Journal of Computer Science and Network Security 11 (6), 208–217.

Chenguang Yang, Jie Chen, X. T., 2007. Algorithm of marriage in honey bees optimization based on the Nelder-Mead method. In: Tianrui Li, Y. X., Ruan, D. (Eds.), International Conference on Intelligent Systems and Knowledge Engineering. Advances in Intelligent Systems Research. Atlantis Press, Amsterdam, Netherlands, pp. 1–7.

Chong, C. S., Sivakumar, A. I., Low, M. Y. H., Gay, K. L., 2006. A bee colony optimization algorithm to job shop scheduling. In: 38th Winter Conference on Simulation. pp. 1954–1961.

Congying, L., Huanping, Z., Xinfeng, Y., 2011. Particle swarm optimization algorithm for quadratic assignment problem. In: International Conference on Computer Science and Network Technology. Vol. 3. pp. 1728 –1731.

Davidovic, T., Selmic, M., Teodorovic, D., Ramljak, D., 2012. Bee colony optimization for scheduling independent tasks to identical processors. Journal of Heuristics 18 (4), 549–569.

Deep, K., Bansal, J. C., 2008. A socio-cognitive particle swarm optimization for multi-dimensional knapsack problem. In: First International Conference on Emerging Trends in Engineering and Technology. pp. 355–360.

Deng-xu, H., Hua-Zheng, Z., Gui-qing, L., 2011. Glowworm swarm optimization algorithm for solving multi-constrained QoS multicast routing problem. In: Seventh International Conference on Computational Intelligence and Security. pp. 66–70.

Dorigo, M., Stützle, T., 2004. Ant Colony Optimization. MIT Press, Cambridge, MA, USA.

Falcon, R., Almeida, M., Nayak, A., 2011. Fault identification with binary adaptive fireflies in parallel and distributed systems. In: IEEE Congress on Evolutionary Computation. pp. 1359–1366.

Feng, Q., Xiao, Q., 2011. Bacterial evolutionary route planning for unmanned aerial vehicle. In: 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce. pp. 3808–3811.

Feng, X., Lau, F. C. M., Gao, D., 2009. A new bio-inspired approach to the traveling salesman problem. In: Zhou, J. (Ed.), Complex Sciences. Vol. 5 of Lecture Notes of the Institute of Computer Sciences. Springer, Berlin, Germany, pp. 1310–1321.

Fister, I., Yang, X.-S., Fister, I., Brest, J., 2012. Memetic firefly algorithm for combinatorial optimization. In: Fifth International Conference on Bioinspired Optimization Methods and their Applications. pp. 75–86.

Garnier, S., Gautrais, J., Theraulaz, G., 2007. The biological principles of swarm intelligence. Swarm Intelligence 1 (1), 3–31.

Gherboudj, A., Layeb, A., Chikhi, S., 2012. Solving 0-1 knapsack problems by a discrete binary version of cuckoo search algorithm. International Journal of Bio-Inspired Computation 4 (4), 229–236.

Glover, F., Laguna, M., 1997. Tabu Search. Kluwer Academic, Norwell, MA, USA.

Havens, T. C., Spain, C. J., Salmon, N. G., Keller, J. M., 2008. Roach infestation optimization. In: IEEE Swarm Intelligence Symposium. IEEE Press, Piscataway, USA, pp. 1–7.

He, D., Qu, L., Guo, X., 2009. Artificial fish-school algorithm for integer programming. In: International Conference on Information Engineering and Computer Science. pp. 1–4.

Hembecker, F., Lopes, H. S., Godoy, Jr., W., 2007. Particle swarm optimization for the multidimensional knapsack problem. In: Proceedings of the 8th international conference on Adaptive and Natural Computing Algorithms, Part I. Vol. 4431 of Lecture Notes in Computer Science. Springer-Verlag, Heidelberg, pp. 358–365.

Hendrawan, Y., Murase, H., 2011. Neural-discrete hungry roach infestation optimization to select informative textural features for determining water content of cultured sunagoke moss. Environment Control in Biology 49 (1), 1–21.

Hernández, H., Blum, C., 2012. Distributed graph coloring: an approach based on the calling behavior of Japanese tree frogs. Swarm Intelligence 6 (2), 117–150.

Inoue, T., Furuhashi, T., Maeda, H., Takaba, M., 2000. A study on bacterial evolutionary algorithm engine for interactive nurse scheduling support system. In: 26th Annual Conference of the IEEE Industrial Electronics Society. Vol. 1. pp. 651–656.

Karaboga, D., 2005. An idea based on honey bee swarm for numerical optimization. Tech. rep., Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboga, D., Gorkemli, B., 2011. A combinatorial artificial bee colony algorithm for traveling salesman problem. In: International Symposium on Intelligent Systems and Applications. IEEE Press, pp. 50–53.

Kennedy, J., Eberhart, R., 2001. Swarm Intelligence. Morgan Kaufmann, San Francisco, USA.

Kennedy, J., Eberhart, R. C., 1997. A discrete binary version of the particle swarm algorithm. IEEE International Conference on Computational Cybernetics and Simulation 5, 4104–4108.

Koudil, M., Benatchba, K., Tarabet, A., Sahraoui, E. B., 2007. Using artificial bees to solve partitioning and scheduling problems in codesign. Applied Mathematics and Computation 186 (2), 1710–1722.

Krishnanand, K. N., Ghose, D., 2005. Detection of multiple source locations using a glowworm metaphor with applications to collective robotics. In: Proceedings of the IEEE Swarm Intelligence Symposium. pp. 84–91.

Kumar, A., Chakarverty, S., 2011. Design optimization for reliable embedded system using cuckoo search. In: 3rd International Conference on Electronics Computer Technology. Vol. 1. pp. 264 –268.

Labed, S., Gherboudj, A., Chikhi, S., 2011. A modified hybrid particle swarm optimization algorithm for multidimensional knapsack problem. International Journal of Computer Applications 34 (2), 11–16.

Layeb, A., 2011. A novel quantum inspired cuckoo search for knapsack problems. International Journal of Bio-Inspired Computation 3, 297–305.

Li, X., Wang, J., Zhou, J., Yin, M., 2010. An effective GSA based memetic algorithm for permutation

flow shop scheduling. In: IEEE Congress on Evolutionary Computation. pp. 1–6.

Li, Z., Li, N., 2009. A novel multi-mutation binary particle swarm optimization for 0/1 knapsack problem. In: 21st Annual International Conference on Chinese Control and Decision Conference. pp. 3090–3095.

Lin, S., B, Kernighan, W., 1973. An effective heuristic algorithm for the traveling-salesman problem. Operations Research 21 (2), 498–516.

Lin, T.-L., Horng, S.-J., Kao, T.-W., Chen, Y.-H., Run, R.-S., Chen, R.-J., Lai, J.-L., Kuo, I.-H., 2010. An efficient job-shop scheduling algorithm based on particle swarm optimization. Expert Systems with Applications 37 (3), 2629–2636.

Lopes, H. S., Coelho, L. S. ., 2005. Particle swarn optimization with fast local search for the blind traveling salesman problem. International Conference on Hybrid Intelligent Systems, 245–250.

Lucic, P., Teodorovic, D., 2002. Transportation modeling: an artificial life approach. In: 14th IEEE International Conference on Tools with Artificial Intelligence. pp. 216–223.

Marinakis, Y., Marinaki, M., Matsatsinis, N., 2008. A hybrid clustering algorithm based on honey bees mating optimization and greedy randomized adaptive search procedure. In: Maniezzo, V., Battiti, R., Watson, J.-P. (Eds.), Learning and Intelligent Optimization. Springer-Verlag, Heidelberg, Germany, pp. 138–152.

Marinakis, Y., Marinaki, M., Matsatsinis, N. F., 2009. A hybrid discrete artificial bee colony-GRASP algorithm for clustering. In: International Conference on Computers & Industrial Engineering. pp. 548–553.

Miwa, M., Inoue, T., Matsuzaki, M., Furuhasi, T., Okuwa, S., 2002. Nurse scheduling system using bacterial evolutionary algorithm hardware. In: 28th Annual Conference of the IEEE Industrial Electronics Society. Vol. 3. pp. 1801–1805.

Mladenovic, N., Hansen, P., 1997. Variable neighborhood search. Computers and Operations Research 24, 1097–100.

Nakamura, R. Y. M., Pereira, L. A. M., Costa, K. A., Rodrigues, D., Papa, J. P., Yang, X.-S., 2012. BBA: a binary bat algorithm for feature selection. In: Conference on Graphics, Patterns and Images. IEEE Computer Society, Los Alamitos, USA, pp. 291–297.

Nawa, N., Furuhashi, T., 1999. Fuzzy system parameters discovery by bacterial evolutionary algorithm. IEEE Transactions on Fuzzy Systems 7 (5), 608–616.

Ozbakir, L., Baykasoğlu, A., Tapkan, P., 2010. Bees algorithm for generalized assignment problem. Applied Mathematics and Computation 215 (11), 3782–3795.

Pacurib, J. A., Seno, G. M. M., Yusiong, J. P. T., 2009. Solving Sudoku puzzles using improved artificial bee colony algorithm. In: Fourth International Conference on Innovative Computing, Information and Control. pp. 885–888.

Palit, S., Sinha, S. N., Molla, M. A., Khanra, A., Kule, M., 2011. A cryptanalytic attack on the knapsack cryptosystem using binary firefly algorithm. In: 2nd International Conference on Computer and Communication Technology. pp. 428–432.

Pan, Q.-Q., Tasgetiren, M. F., Liang, Y.-C., 2008. A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem. Computers & Operations Research 35 (9), 2807 – 2839.

Papa, J. P., Pagnin, A., Schellini, S. A., Spadotto, A., Guido, R. C., Ponti, M., Chiachia, G., Falcao, A. X., 2011. Feature selection through gravitational search algorithm. In: IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 2052–2055.

Parpinelli, R. S., Lopes, H. S., 2011. New inspirations in swarm intelligence: a survey. International Journal of Bio-Inspired Computation 3 (1), 1–16.

Passino, K. M., 2002. Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Systems Magazine 22 (3), 52–67.

Pham, D., Koc, E., Lee, J., Phrueksanant, J., 2007a. Using the bees algorithm to schedule jobs for a machine. In: Proceedings of Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance. pp. 430–439.

Pham, D. T., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S., Zaidi, M., 2005. The bees algorithm. Technical note, Manufacturing Engineering Center, Cardiff University, UK.

Pham, D. T., Otri, S., Afify, A., Mahmuddin, M., Al-Jabbouli, H., 2007b. Data clustering using the bees algorithm. In: 40th CIRP International Seminar on Manufacturing Systems. p. s.p.

Pulikanti, S., Singh, A., 2009. An artificial bee colony algorithm for the quadratic knapsack problem. In: The 16th International Conference on Neural Information Processing. Vol. 5864 of Lecture Notes in Computer Science. Springer-Verlag, Heidelberg, Germany, pp. 196–205.

Rabanal, P., Rodríguez, I., Rubio, F., 2007. Using river formation dynamics to design heuristic algorithms. In: Akl, S. G., Calude, C. S., Dinneen, M. J., Rozenberg, G., Wareham, T. (Eds.), Unconventional Computation. Vol. 4618 of Lecture Notes in Computer Science. Springer, Heidelberg, Germany, pp. 163–177.

Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S., 2009. GSA: A gravitational search algorithm. Information Sciences 179 (13), 2232–2248.

Rosendo, M., Pozo, A., 2010. Applying a discrete particle swarm optimization algorithm to combinatorial problems. In: Brazilian Symposium on Neural Networks. IEEE Computer Society, Los Alamitos, CA, USA, pp. 235–240.

Sayadi, M. K., Ramezanian, R., Ghaffari-Nasab, N., 2010. A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. International Journal of Industrial Engineering Computations 1, 1–10.

Shah-Hosseini, H., 2007. Problem solving by intelligent water drops. In: IEEE Congress on Evolutionary Computation. pp. 3226–3231.

Shen, X., Li, Y., Chen, C., Yang, J., Zhang, D., 2012. Greedy continuous particle swarm optimisation algorithm for the knapsack problems. International Journal of Computer Applications in Technology 44 (2), 137–144.

Singh, V., Singh, D., Tiwari, R., Shukla, A., 2011. RGBCA-genetic bee colony algorithm for travelling salesman problem. In: World Congress on Information and Communication Technologies. IEEE Press, pp. 1002–1008.

Tasgetiren, M., Liang, Y., Sevkli, M., Gencyilmaz, G., 2004a. Particle swarm optimization algorithm for makespan and maximum lateness minimization in permutation flowshop sequencing problem. In: 4th International Symposium on Intelligent Manufacturing Systems. pp. 237–250.

Tasgetiren, M., Sevkli, M., Liang, Y.-C., Gencyilmaz, G., 2004b. Particle swarm optimization

algorithm for single machine total weighted tardiness problem. In: IEEE Congress on Evolutionary Computation. Vol. 2. pp. 1412–1419.

Tasgetiren, M. F., Suganthan, P. N., Pan, Q.-Q., 2007. A discrete particle swarm optimization algorithm for the generalized traveling salesman problem. In: 9th Annual Conference on Genetic and Evolutionary Computation. ACM Press, New York, USA, pp. 158–165.

Teo, J., Abbass, H. A., 2001. An annealing approach to the mating-flight trajectories in the marriage in honey bees optimization algorithm. Technical Report CA04/01, School of Computer Science, University of New South Wales at ADFA, Canberra, Australia.

Teodorovic, D., Dell'Orco, M., 2005. Bee colony optimization – a cooperative learning approach to complex transportation problems. In: 16th Mini-Euro Conference on Advanced OR and AI Methods in Transportation. pp. 51–60.

Ucar, H., Tasgetiren, M. F., 2006. A particle swarm optimization algorithm for permutation flow shop sequencing problem with the number of tardy jobs criterion. In: 5th International Symposium on Intelligent Manufacturing Systems. pp. 237–250.

Verma, R., Kumar, S., 2012. DSAPSO: DNA sequence assembly using continuous particle swarm optimization with smallest position value rule. In: 1st International Conference on Recent Advances in Information Technology. pp. 410–415.

Wan, N. F., Nolle, L., 2009. Solving a multi-dimensional knapsack problem using hybrid particle swarm optimization algorithm. In: 23rd European Conference on Modelling and Simulation.

Wong, L.-P., Low, M. Y., Chong, C. S., 2008a. A bee colony optimization algorithm for traveling salesman problem. In: Second Asia International Conference on Modelling & Simulation. IEEE Press, pp. 818–823.

Wong, L.-P., Puan, C. Y., Low, M. Y. H., Chong, C. S., 2008b. Bee colony optimization algorithm with big valley landscape exploitation for job shop scheduling problems. In: Proceedings of the 40th Winter Simulation Conference. pp. 2050–2058.

Xiangyang, S., Minghao, Z., Yamin, Z., 2011. Improved artificial fish school algorithm to solve knapsack problem. Computer Engineering and Applications 47 (21), 43.

Xu, S., Ji, Z., Pham, D. T., Yu, F., 2010. Bio-inspired binary bees algorithm for a two-level distribution optimisation problem. Journal of Bionic Engineering 7, 161–167.

Yang, X. S., 2008. Firefly algorithm. Nature-inspired Metaheuristic Algorithms. Luniver Press, Bristol, UK, Ch. 8.

Yang, X. S., 2009. Firefly algorithm, Lévy flights and global optimization. XXVI Research and Development in Intelligent Systems. Springer, London, UK, pp. 209–218.

Yang, X. S., 2010. A new metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization. Vol. 284 of Studies in Computational Intelligence. Springer, Berlin, pp. 65–74.

Yang, X.-S., Deb, S., 2009. Cuckoo search via Lévy flights. World Congress on Nature & Biologically Inspired Computing, 210–214.

Yousif, A., Abdullah, A. H., Nor, S. M., Abdelaziz, A. A., 2011. Scheduling jobs on grid computing using firefly algorithm. Journal of Theoretical and Applied Information Technology 33 (2), 155–164.