Yugoslav Journal of Operations Research
25 (2015), Number 1, 33–56
DOI: 10.2298/YJOR131011017D

Invited survey

BEE COLONY OPTIMIZATION PART I: THE ALGORITHM OVERVIEW

Tatjana DAVIDOVIĆ
Mathematical Institute, Serbian Academy of Sciences and Arts
tanjad@mi.sanu.ac.rs

Dušan TEODOROVIĆ
Faculty of Transport and Traffic Engineering, University of Belgrade
dusan@sf.bg.ac.rs

Milica ŠELMIĆ
Faculty of Transport and Traffic Engineering, University of Belgrade
m.selmic@sf.bg.ac.rs

Received: October 2013 / Accepted: May 2014

Abstract: This paper is an extensive survey of the Bee Colony Optimization (BCO) algorithm, proposed for the first time in 2001. BCO and its numerous variants belong to a class of nature-inspired meta–heuristic methods, based on the foraging habits of honeybees. Our main goal is to promote it among the wide operations research community. BCO is a simple, but efficient meta–heuristic technique that has been successfully applied to many optimization problems, mostly in transport, location and scheduling fields. Firstly, we shall give a brief overview of the meta–heuristics inspired by bees' foraging principles, pointing out the differences between them. Then, we shall provide the detailed description of the BCO algorithm and its modifications, including the strategies for BCO parallelization, and give the preliminary results regarding its convergence. The application survey is elaborated in Part II of our paper.

Keywords: Meta–heuristics, Swarm Intelligence, Foraging of Honey Bees.

MSC: 68T20, 90C59, 92D50.

1. INTRODUCTION

The nature-inspired algorithms are motivated by a variety of biological and natural processes. Their popularity is based primarily on the ability of biological systems to efficiently adapt to frequently changeable environments. Evolutionary computation, neural networks, ant colony optimization, particle swarm optimization, artificial immune systems, and bacteria foraging algorithm are among the algorithms and concepts that were motivated by nature.

Swarm behavior is one of the main features of different colonies of social insects (bees, wasps, ants, termites). This type of behavior is principally characterized by autonomy, distributed functioning, and self-organizing. Swarm Intelligence [5, 6] is the area of Artificial Intelligence based on studying actions of individuals in various decentralized systems. When creating Swarm Intelligence models and techniques, researchers apply some principles of the natural swarm intelligence.

In the last two decades, the researchers have been studying the behavior of social insects in an attempt to utilize the swarm intelligence concept and build up various artificial systems. Here are some optimization algorithms inspired by bees' behavior that appeared during the last decade: Bee System [27, 41], Bee Colony Optimization (BCO) [50], Marriage in Honey-Bees Optimization (MBO) [1], BeeHive [55], Honey Bees [34], Artificial Bee Colony (ABC) [22], Bee System Optimization (BSO) [16], Bees Algorithm [39, 40], Honey Bee Marriage Optimization (HBMO) [2], Fast Marriage in Honey Bees Optimization (MHBO) [58], Virtual Bee Algorithm (VBA) [59]. These algorithms apply *information share models* to beat restrictions on the applicability of optimization techniques. In all these approaches, there are few agents who search solution space at the same time. Artificial bees (agents) in all considered approaches have incomplete information when solving the problem. There is no global control in any of these approaches. Artificial bees are based on the concept of cooperation. Cooperation enables bees to be more efficient, sometimes even to achieve goals they could not achieve individually. These algorithms denote general algorithmic frameworks that could be applied to diverse optimization problems. The excellent surveys of the algorithms inspired by bees' behavior in nature are given in [23, 48, 49], while in [24] a comprehensive survey of ABC and its applications is presented. Here, we focus on BCO with the main goal to describe basic concept of the algorithm proposed in [27] (under the name Bee System, renamed to BCO in [50]) and to illustrate its evolution. BCO is a meta–heuristic method, since it represents a general algorithmic framework applicable to various optimization problems in management, engineering, and control that could always be tailored for a specific problem. BCO belongs to the class of population-based algorithms. Complex initial formulation of the algorithm has been evolving to simpler versions through numerous applications [12, 14, 15, 32, 34, 36, 37, 47, 51, 52, 56, 57]. Part II of this paper is devoted to detailed survey of BCO applications.

The BCO algorithm underwent numerous changes trough the process of evolution from its development in 2001 until nowadays. Moreover, in order to solve hard optimization problems, the initial constructive BCO was modified and a new

concept based on the improving complete solutions was developed. This concept enabled obtaining better final solutions than the ones resulted from constructive moves only.

In addition, parallel variants of the BCO algorithm were developed. The main goal of the parallelization, in general, is to speed up the computations needed to solve a particular problem by engaging several processors and dividing the total amount of work between them. When meta–heuristics are under consideration, the performance of parallelization strategy is influenced also by the quality of final solution. Namely, meta–heuristics represent stochastic search procedures (and BCO is not an exception) that may not result in the same solution even in repeated sequential executions. On the other hand, parallelization may assure the extension of the search space that could yield either improvement or degradation of the final solution quality. Therefore, the quality of final solution should also be considered as a parameter of parallelization strategy performance. Consequently, the combination of gains may be expected: parallel execution can enable efficient search of different regions in the solution space yielding the improvement of the final solution quality within smaller amount of execution time.

BCO has proven to be suitable method for solving non-standard combinatorial optimization problems, e.g., those containing inaccurate data or involving optimization according to multiple criteria. In these cases, the application of BCO requires it to be hybridized by the appropriate techniques.

As for many other meta–heuristic methods, the quality of the final solution cannot be evaluated with respect to the optimal one. However, some theoretical aspects connected to the asymptotic convergence could be considered. Here we recall some results identifying necessary conditions for the theoretical verification of the BCO algorithm.

This paper presents a brief description of the meta–heuristics inspired by bees' foraging principles, details of the BCO algorithm, as well as of the algorithm changes through its evolution. The rest of the paper is organized as follows. Biological background is presented in Section 2. The survey of all algorithms that rely on foraging habits of honeybees is given in Section 3. BCO is described in Section 4. Modifications of the proposed algorithm are given in Section 5. Section 6 contains the theoretical aspects of BCO convergence, and the last section is devoted to conclusions and possible directions for a future research.

2. BIOLOGICAL BACKGROUND

Swarm behavior (fish schools, flocks of birds, herds of land animals, insects' communities, etc.) is based on the *biological needs* of individuals to stay together. In such a way, individuals increase the probability to stay alive since predators usually attack only the isolated individuals. Flocks of birds, herds of animals, and fish schools are characterized by collective movement. Colonies of various social insects (bees, wasps, ants, termites) are also characterized by swarm behavior. Swarm behavior is primarily characterized by autonomy, distributed functioning and self-organizing. The communication systems between individual insects

contribute to the *collective intelligence* pattern named "Swarm Intelligence" in [5, 6].

Swarm Intelligence represents the branch of the Artificial Intelligence that investigates individuals' actions in different decentralized systems. These decentralized systems (Multi Agent Systems) are composed of physical individuals (robots, for example) or "virtual" (artificial) ones that communicate, cooperate, collaborate, exchange information and knowledge, and perform some tasks in their environment. When designing Swarm Intelligence models, researchers use some principles of the natural swarm intelligence. The development of artificial systems does not usually involve the entire imitation of natural systems, but explores and adapts them while searching for ideas and models.

Bees in nature look for food by exploring the fields in the neighborhood of their hive. They collect and accumulate the food for later use by other bees. Typically, in the initial step, some scouts search the region. Completing the search, scout bees return to the hive and inform their hive-mates about the locations, quantity, and quality of the available food sources in the areas they have examined. In case they have discovered nectar in the previously investigated locations, scout bees dance in the so-called "dance floor area" of the hive, in an attempt to "advertise" food locations and encourage the remaining members of the colony to follow their lead. The information about the food quantity is presented using a ritual called a "waggle dance". If a bee decides to leave the hive and collect the nectar, it will follow one of the dancing scout bees to the previously discovered patch of flowers. Upon arrival, the foraging bee takes a load of nectar and returns to the hive relinquishing the nectar to a food store. Several scenarios are then possible for a foraging bee: (1) it can abandon the food location and return to its role of an uncommitted follower; (2) it can continue with the foraging behavior at the discovered nectar source without recruiting the rest of the colony; (3) it can try to recruit its hive-mates with the dance ritual before returning to the food location. The bee opts for one of the above alternatives. As several bees may be attempting to recruit their hive-mates on the dance floor area at the same time, it is unclear how an uncommitted bee decides which recruiter to follow. The only obvious fact is that "the loyalty and recruitment among bees are always a function of the quantity and quality of the food source" [7]. The described process continues repeatedly, while the bees at a hive accumulate nectar and explore new areas with a potential food sources.

3. THE ALGORITHMS BASED ON FORAGING HABITS OF HONEYBEES

Numerous algorithms have appeared in the recent literature as the paradigm based on foraging habits of honeybees [3, 8, 11, 12, 13, 14, 18, 27, 32, 34, 37, 43, 44, 46, 47, 50, 53, 52, 54, 55, 56, 57]. They were mainly developed starting from one of the following methods ABC [22], Bees Algorithm [39, 40] and Bee System [27]. Different authors propose various models of implementations and verify them on numerous combinatorial optimization problems.

Main steps of any algorithm based on foraging habits of honeybees are: *foraging* and *waggle dancing*. Foraging is the solution generation phase, while the role of the waggle dance (the information exchange phase) is to examine quality of the existing solutions and direct the generation to the new ones. The idea for the development of these algorithms was based on the simple rules for modeling the organized nectar collection.

These algorithms use a similarity between the way in which bees in nature look for food and the way in which optimization algorithms search for an optimum of the combinatorial optimization problems. The main idea was to create the multi agent system (the colony of artificial bees) capable to efficiently solve hard combinatorial optimization problems. The artificial bees explore through the search space looking for the feasible solutions. In order to increase the quality of discovered solutions, artificial bees cooperate and exchange information. Via collective knowledge and information exchange, the artificial bees focus on more promising areas and gradually discard solutions from the less promising ones.

Among the first algorithms based on foraging concept is the one proposed in [27]. The authors initially named their algorithm Bee System, but starting with paper [50], the name BCO has been used. The aim of this paper is to propagate this BCO algorithm by describing basic steps and its evolution toward simpler but more efficient method. Before that, in the rest of this section, we briefly summarize various algorithms based on foraging habits of honeybees, pointing out the differences between them, and listing their applications.

Bulk of the papers proposing algorithms based on foraging habits of honeybees and their applications to numerous combinatorial optimization problems appeared in the last couple of years [3, 8, 11, 12, 13, 14, 18, 22, 26, 27, 28, 29, 30, 32, 34, 36, 37, 43, 44, 46, 47, 50, 51, 53, 52, 54, 55, 56, 57]. Different implementations of basic steps (foraging and waggle dancing) are the reasons for such a variety of algorithms. As the major differences between these algorithms we point out the following:

- various biological inspiration sources, mainly [7] or [42];
- the initial solutions are generated in various ways: randomly [52, 54] or in some constructive, probability based manner [12];
- the number of bees may vary during the search process [27] and bees may have different role (e.g., scouts, workers and onlookers in [22, 54]), while in [12, 13, 14, 32, 47, 53] the number of bees is fixed and they perform the same algorithm steps;
- in some of the algorithms, whole solutions are propagated [12, 56] and in others, partial solutions [3, 14, 32, 47] or even the solution components [46] influence the decision making process;
- dance duration, i.e., the number of iterations a solution is propagated, changes in [8, 34], while in [12, 13, 14] only the solutions from the current iteration are propagated;

- some concepts involve solution improvement by applying e.g., local search [54, 56] or fuzzy logic [51];
- different concepts for solution modification: constructive [15, 47], and improvement concept [12, 36, 52].

The algorithms based on foraging habits of honeybees have been applied to various optimization problems in location analysis [3, 12, 15, 17, 47, 53], biosciences [18, 46], economy [54], scheduling [13, 34, 43], transport and engineering [32, 37, 51, 56, 57], medicine [52], etc.

4. THE BCO META–HEURISTIC

In the period 1999-2003, the basic concepts of BCO [27, 28, 29, 30] were introduced under the name Bee System, by Dušan Teodorović (adviser) and Panta Lučić (Ph.D. candidate) while doing research at Virginia Tech. BCO is a nature–inspired meta–heuristic method developed for efficiently finding solutions to difficult combinatorial optimization problems.

The basic idea behind BCO is to build the multi agent system (colony of artificial bees) that will search for good solutions of various combinatorial optimization problems, exploring the principles used by honey bees during nectar collection process. Artificial bee colony usually consists of a small number of individuals, but nevertheless, BCO principles are gathered from the natural systems. Artificial bees investigate through the search space looking for the feasible solutions. In order to find the best possible solutions, autonomous artificial bees collaborate and exchange information. Using collective knowledge and information sharing, artificial bees concentrate on the more promising areas and slowly abandon solutions from the less promising ones. Step by step, artificial bees collectively generate and/or improve their solutions. The BCO search is running in iterations until some predefined stopping criterion is satisfied.

During the evolution of the BCO algorithm, the authors have developed two different approaches. The first approach [13, 47] is based on constructive steps in which bees build solutions piece by piece. The second, a very actual approach of the BCO algorithm [12, 37, 52], is based on the improvement of complete solutions in order to obtain the best possible final solution. This approach is named BCOi. In the text to follow, we explain both concepts through the general description of the algorithm.

The population of agents consisting of B individuals (artificial bees) is engaged in BCO. Each artificial bee is responsible for one solution of the problem. There are two alternating phases (*forward pass* and *backward pass*) constituting a single step in the BCO algorithm. In each forward pass, all artificial bees explore the search space. They apply a predefined number of moves, which construct/improve the partial/complete solutions, yielding the new partial/complete solutions.

For example, let bees Bee 1, Bee 2, . . . , Bee B participate in the decision–making process on n entities. The entity could be a subset (one or more) of components of the partial solution according to the constructive version, or a single complete

solution that is going to be enhanced in the following algorithm phases according to BCOi. The possible situation after s-th forward pass is illustrated in Fig. 1. Rectangles in Fig. 1 represent partial/complete solutions associated to the bees. Different patterns denote that to each bee is associated a different solution.

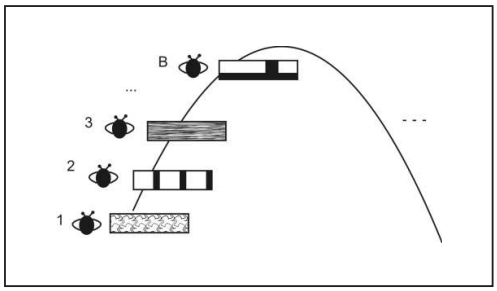

Figure 1: An illustration of partial/complete solutions after s-th forward pass

Having obtained new partial/complete solutions, the second phase, the so-called backward pass starts. In the backward pass, all artificial bees share the information about the quality of their partial/complete solutions. In nature, bees would return to the hive, perform a dancing ritual ("waggle dance") that would inform other bees about the amount of food they have discovered, and the proximity of the patch to the hive. In the search algorithm, the announcement of the solution quality is performed by the calculation of the objective function value for each partial/complete solution. Having all solutions evaluated, each bee decides with a certain probability whether it will stay *loyal* to its solution or not. The bees with better solutions have more chances to keep and advertise their solutions. Contrary to the bees in nature, artificial bees that are loyal to their partial/complete solutions are at the same time the *recruiters*, i.e., their solutions would be considered by other bees. Once the solution is abandoned, the corresponding bee becomes *uncommitted* and has to select one of the advertised solutions. This decision is taken with a probability, such that better advertised solutions have greater opportunities to be chosen for further exploration. In such a way, within each backward pass, all bees are divided into two groups (R recruiters, and remaining $B - R$ uncommitted bees) as it is shown in Fig. 2. Values for R and $B - R$ are changing from one backward pass to another.

Let us assume that after comparing all generated partial/complete solutions, Bee 1, from Fig. 1 decided to abandon its own solution and to join Bee B (see Fig. 3). Bee 1 and Bee B "fly together" along the path already generated by the Bee B. In practice, this means that partial/complete solution generated by Bee B is associated (copied) to Bee 1 also (rectangles which denote solutions associated to Bee 1 and Bee B are the same in Fig. 3). After that, both Bee 1 and Bee B are free to make their individual decisions about the next step to be made. The Bees 2, 3, from previous example, keep already generated partial/complete solutions.

In the next forward pass, according to the constructive BCO, each bee adds

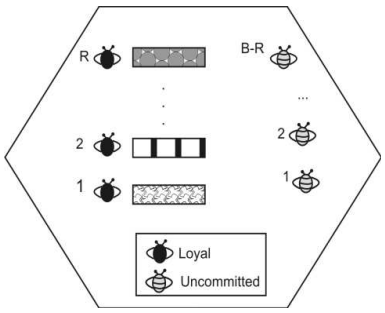

Figure 2: Recruiting of uncommitted followers

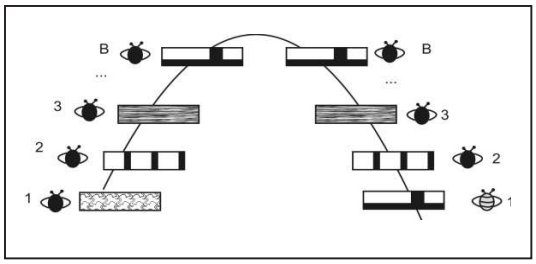

Figure 3: The possible result of a recruiting process in s-th backward pass

new (different) components to the previously generated partial solution, while in BCOi bees modify some components of the complete solutions in order to generate higher quality ones. The described situation after the next forward pass is illustrated in Fig. 4 (bees' partial/complete solutions are changed, and it is shown symbolically by the associated rectangles with new patterns).

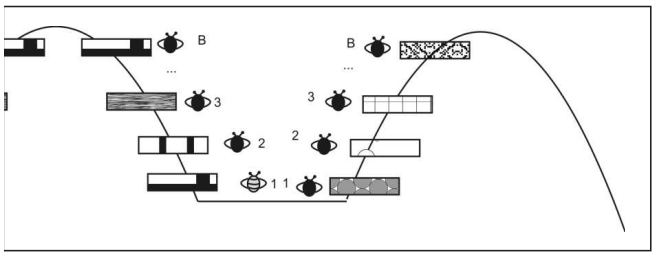

Figure 4: An example of partial/complete solutions after (s+1)-th forward pass

The two phases of the search algorithm, namely the forward and backward pass, alternate NC times, i.e., until each bee completes the generation of its solution

or performs *NC* solution modifications. In former (constructive) variants of the BCO algorithm, parameter *NC* was used to count the number of components to be added to partial solutions during each forward pass. Consecutively, the number of forward/backward passes was calculated as a function of *NC*. In order to unify the algorithm description, we changed the original meaning of *NC* after the development of BCOi. Some other modifications of BCO parameters can also be found in recent literature [36, 37]. In any case, *NC* is a parameter used to define the frequency of information exchange between bees. When *NC* steps are completed, the best among all *B* solutions is determined. It is then used to update global best solution and an iteration of BCO is accomplished. At this point, all *B* solutions are deleted and the new iteration could start. The BCO algorithm runs iteration by iteration until a stopping condition is met. The possible stopping condition could be, for example, the maximum number of iterations, the maximum number of iterations without the improvement of the objective function value, maximum allowed CPU time, etc. At the end, the best found solution (the so called current global best) is reported as the final one.

The BCO algorithm parameters whose values need to be set prior the algorithm execution are:

B — the number of bees involved in the search and

NC — the number of forward (backward) passes constituting a single BCO iteration.

The pseudo-code of the BCO algorithm is given in Fig. 5.

Steps (1), (a) and (b) are problem dependent and should be resolved in each BCO implementation. On the other hand, there are formulae specifying steps (c), loyalty decision, and (d), recruiting process, and they are described in the rest of this section.

4.1. Loyalty Decision

After the completion of a forward pass, each bee decides whether to stay loyal to the previously discovered solution or not. This decision depends on the quality of its own solution related to all other existing solutions. The probability that *b*-th bee (at the beginning of the new forward pass) is loyal to its previously generated partial/complete solution is expressed as follows:

$$p_b^{u+1} = e^{-\frac{O_{\max}-O_b}{u}}, \quad b = 1, 2, \ldots, B \tag{1}$$

where:
O_b - denotes the normalized value for the objective function of partial/complete solution created by the *b*-th bee;
$O_{\max}$ - represents maximum over all normalized values of partial/complete solutions to be compared;
u - counts the forward passes (taking values 1, 2, . . . , *NC*).

The normalization is performed in two ways, depending on whether a minimization or maximization of the objective function is required. If C_b (b =

Initialization: Read problem data, parameter values (B and NC),
and stopping criterion.
Do
(1) Assign a(n) (empty) solution to each bee.
(2) *For* ($i = 0; i < NC; i++$)
//**forward pass**
(a) *For* ($b = 0; b < B; b++$)
For ($s = 0; s < f(NC); s++$)//count moves
(i) Evaluate possible moves;
(ii) Choose one move using the roulette wheel;
//**backward pass**
(b) *For* ($b = 0; b < B; b++$)
Evaluate the (partial/complete) solution of bee b;
(c) *For* ($b = 0; b < B; b++$)
Loyalty decision for bee b;
(d) *For* ($b = 0; b < B; b++$)
If (b is uncommitted), choose a recruiter by the roulette wheel.
(3) Evaluate all solutions and find the best one. Update x_{best} and $f(x_{best})$
while stopping criterion is not satisfied.
return (x_{best}, $f(x_{best})$)

Figure 5: Pseudo-code for BCO

$1, 2, \ldots, B$) denotes the objective function value of b-th bee partial/complete solution, normalized value of the C_b in the case of minimization is calculated as follows:

$$O_b = \frac{C_{\max} - C_b}{C_{\max} - C_{\min}}, \quad b = 1, 2, \ldots, B \tag{2}$$

where C_{min} and C_{max} are the values of partial/complete solutions related to minimal and maximal objective function value, respectively, obtained by all engaged bees. From equation (2), it could be seen that if b-th bee partial/complete solution is closer to maximal value of all obtained solutions, C_max, than its normalized value, O_b, is smaller and vice versa.

In the case of maximization criterion, normalized value of Cb is calculated as follows:

$$O_b = \frac{C_b - C_{\max}}{C_{\max} - C_{\min}}, \quad b = 1, 2, \ldots, B \tag{3}$$

From equation (3), it is obvious that if the value of the partial/complete solution C_b, is higher, then its normalized value O_b, is larger, and vice versa.

Using equation (1) and a random number generator, each artificial bee decides whether to become uncommitted follower or to continue exploring its own solution. If chosen random number is smaler than the calculated probability, then the

bee stays loyal to its own solution. Otherwise, if the random number is greater than the probability p_b^{u+1}, the bee becomes uncommitted.

Let us discuss equation (1) in some more details. A greater O_b value corresponds to a better generated solution, and a higher probability of the bee staying loyal to the previously discovered solution. The higher index in the forward pass increases the influence of the already discovered solution. This is expressed by the term u in the denominator of the exponent (equation (1)). In other words, at the beginning of the search process bees are "braver" when searching the solution space. The more forward passes are made, the less courage they have: as we approach to the end of the search process, the bees are more focused on the already known solutions.

Some other probability functions are examined in [31], indicating that alternative ways to determine loyalty should also be considered since for some combinatorial problems they may result in faster execution of the BCO algorithm.

4.2. Recruiting Process

For each uncommitted bee, it is decided which recruiter it will follow, taking into account the quality of all advertised solutions. The probability that b's partial/complete solution would be chosen by any uncommitted bee equals:

$$p_b = \frac{O_b}{\sum_{k=1}^{R} O_k}, \quad b = 1, 2, \ldots, R \tag{4}$$

where O_k represents the normalized value for the objective function of the k-th advertised solution, and R denotes the number of recruiters. Using equation (4) and a random number generator, each uncommitted follower joins one recruiter through a roulette wheel.

The roulette wheel is a well-known model of choice. The main inspiration for its development came from a game-gambling roulette. Any solution can be chosen, and the probability of its selection (the size of a particular slot on the roulette wheel) depends on the quality of that solution, i.e., the value of the objective function.

In practice, the size of the slot on the roulette wheel associated to each solution is determined by the ratio of the corresponding normalized objective function value and the sum of the normalized objective function values for all advertised solutions. On the one hand, a solution with better objective function value has a higher chance to be selected. On the other hand, there is still a possibility that it will be eliminated from further search process.

5. THE MODIFICATIONS OF THE BCO ALGORITHM

The BCO algorithm underwent numerous changes trough the process of evolution from its appearance until nowadays. The first versions of BCO [27, 28, 29, 30]

have more similarities with the behavior of bees in the nature than the recent versions of the algorithm. The main difference between these versions is that hive had an important role in the first version. The hive had specified location that could also be changed during the search process. The other difference is that in the first version not all bees were engaged at the beginning of the search process. The scout bees started the search, and at each stage, new bees joined in by the recruiting process. In this initial BCO version, the authors proposed Logit-based model [33] for calculating the probability of choosing next node to be visited, while in recent versions roulette wheel is used for this purpose.

During numerous applications of the BCO algorithm, it was observed that the constructive version cannot successfully solve some combinatorial optimization problems. Therefore, several modifications of the BCO algorithm were developed in order to solve hard optimization problems. In the text to follow, we will explain all these modifications in details.

5.1. Constructive and Improving Alternatives of the BCO Algorithms

In most of the recent applications, the BCO algorithm was constructive [13, 29, 30, 47, 56, 57]. For each bee, a solution was constructed from scratch, step by step, applying some stochastic, problem specific, heuristic rules. Randomness induced by these stochastic construction processes assured diversity of the search. Within each iteration, B solutions were generated and the best of them was used for updating the current global best solution. Next iteration then resulted in B new solutions, among which, we searched for the new global best one.

The first time implementation of the original constructive BCO method in solving the p-center problem was not successful enough. As a consequence, the authors of [12] decided to develop a new concept based on improving the complete solution held by each bee. The new version of the BCO algorithm is named BCOi, as it is mentioned before. In contrast to the constructive version, in BCOi bees are assigned complete solutions at the beginning, and have to modify them through the iterations. The BCOi concept has not been previously used in the relevant literature: all methods based on bees foraging principles were the constructive ones.

The BCOi algorithm implementation proposed in [12] consists of the following five steps. The first step, called preprocessing is performed "off-line". In this step the input data are transformed in order to reduce the time required for all computations performed online. The second step represents generation of the initial complete solutions. In the third, the most significant step, bees modify current solutions through NC forward passes within the single iteration. This step is the key factor that enables reaching the best possible solution quality. Its main role is to assure different treatment for the same solutions held by different bees. Solution modification has to be stochastic and without local search embedded in the process. Steps 4 (solutions comparison mechanism) and 5 (recruitment) are identical to the corresponding steps from the constructive BCO. The last four steps are executed in the real CPU time.

The idea of improving alternatives could be developed in many different ways, and this approach certainly may be very useful for solving difficult combinatorial optimization problems. It has already been explored in the recent literature for solving some other hard optimization problems like network design [37] and satisfiability problem in the logic with approximate conditional probability [44], while for the berth allocation problem [26], hybrid variant was designed.

5.2. Global knowledge

The recent concepts of the BCO algorithm did not involve information exchange between iterations, i.e., there was no global knowledge [12, 15, 47]. The advantage of this approach is that bees have freedom to generate various solutions that lead to diversification of the solution space search. On the other hand, it may cause potentially promising parts of the solution space to be insufficiently explored. Therefore, it may be useful to take into account information exchange between iterations and its amount should be carefully determined. The authors of [14] notice that an excessive increase of information exchange forces bees to concentrate their search on a small part of the solution space. As a result, the BCO algorithm generates final solutions of poor quality. To overcome this problem, the authors of [14] embedded the existence of the global knowledge into the algorithm.

In order to improve their results on the tested benchmark instances, the authors of [14] allowed the relevant information to be available through different iterations, and to be explored by bees during the solution generation process. The appropriate amount of shared information for this problem is determined experimentally. The following modification of the BCO algorithm yielded the greatest performance increase. The current global best objective function value is used by each bee during both forward and backward passes. Within the forward pass, bees would not perform constructive steps yielding the solutions worse than the current global best if they have an alternative. Regarding the backward pass, the same information is used to prevent bees staying loyal to partial solutions inferior to the current global best objective function. Consequently, the probabilities defined by equation (1) for those bees are set to zero, and they automatically become uncommitted. If all bees become uncommitted in this way, the iteration is interrupted and a new one starts without completing the current partial solutions.

According to the global knowledge concept, for a successful implementation of the BCO algorithm, it is necessary to establish the connections between iterations and therefore, to prevent the generation of solutions worse than the current global best.

Similar principle has also been used in [37]. The authors propagated the best known solution through modification process in the later BCO iterations. Among randomly generated initial solutions, the best was set to be the current best known (incumbent) solution. At the beginning of each iteration, the authors took into account the incumbent as well as B solutions created within previous iteration ($B + 1$ solutions in total). In such a way, they enabled global knowledge about current best solution to be shared through search process.

5.3. Parallelization of the BCO algorithm

Swarm intelligence algorithms are created as multi-agent systems, providing a good basis for the parallelization on different levels. High-level parallelization assumes a coarse granulation of tasks and can be applied to iterations of all these methods. Smaller parts of the algorithms usually also contain a lot of independent executions, and are suitable for low-level parallelization. Parallelization on all levels for various bees' algorithms can be found in recent literature [4, 10, 12, 35, 38, 45]. Here we briefly recall the parallelization strategies for BCO proposed in [10, 12], and systematically reviewed in [9].

The authors of [10, 12] addressed the parallelization of BCO for distributed memory multiprocessors systems. They considered the coarse granulation strategy in both synchronous and asynchronous way. Fine-grained parallelization is not suitable for these multi-processors systems, as it was verified in [12]. Three different strategies for parallelization of BCO, two synchronous and one asynchronous, were proposed in [10]. All strategies may be implemented in several different ways; some details are given in the remaining of this subsection.

5.3.1. Independent run of BCO algorithms

Coarse-grained parallelization of BCO in its simplest form presents the independent execution of necessary computations on different processors, as illustrated in Fig. 6. The main goal of this strategy was to speed up the search performed by BCO by dividing total amount of work between several processors.

In [10], it was realized by a reduction of the stopping criterion on each processor. For example, if the stopping criterion is defined as the allowed CPU time (given as a *runtime* value in seconds), the BCO could run in parallel on q processors for $runtime/q$ seconds. Similar rule can be introduced when maximum number of iterations is selected as the stopping criterion. In both cases, each processor has to perform independently sequential variant of BCO, but with the reduced value of the stopping criteria. The BCO parameters (number of bees B and number of forward/backward passes NC) were the same for all BCO processes executing on different processors in order to assure a load balance between all processors. The BCO algorithms running on different processors were differing in the seeds values. This variant of parallelized BCO was named *Distributed BCO* (DBCO). Another way to implement the coarse grained parallelization strategy proposed in [10] was the following: Instead of the stopping criterion, the number of bees could be divided. Namely, if the sequential execution uses B bees for the search, parallel variant executing on q processors would be using B/q bees only. Actually, on each processor, a sequential BCO is running with the reduced number of bees. This variant was referred to as BBCO since the bees were distributed among processors, and again the BCO parameters had the same values for all BCO processes executing on different processors.

The third variant of the independent BCO execution proposed in [10] was obtained by varying values of the BCO parameters and changing stopping criterion. The authors of [10] referred to this strategy as MBCO, and showed that it introduced more diversification in the search process. In this case, the reductions

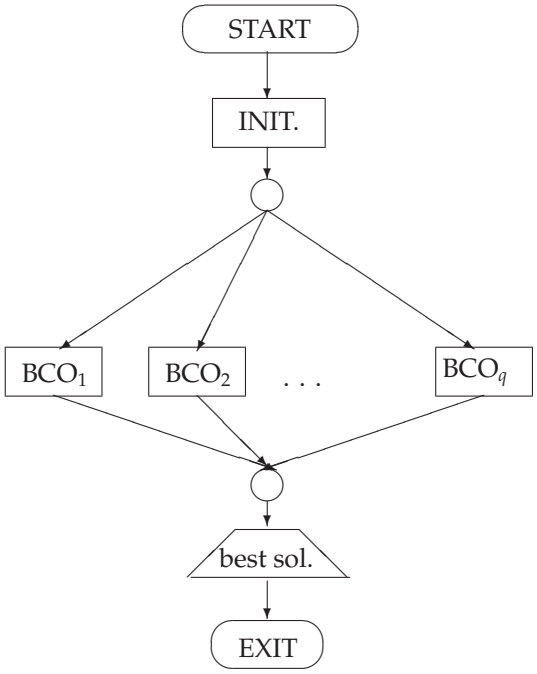

Figure 6: DBCO – Independent execution of several sequential BCO algorithms

required to realize distributed execution in one of the described manner were adjusted in such a way that the resulting load of processors was balanced. For example, if the number of bees on one processor was twice bigger than that on the other processor (and the value for *NC* remained the same), then the stopping criterion of the first process was set to one half of the stopping criterion on the second process.

5.3.2. Synchronous cooperation of BCO algorithms

More sophisticated way to realize coarse-grained parallelization is cooperative work of several BCO processes. At certain predefined execution points, all processes exchange the relevant data (usually the best solutions they obtained by that time) that are used to guide further search. This synchronous strategy was proposed in [10] and named *Cooperative BCO* (CBCO). The block-diagram illustrating this strategy would be similar to the one presented in Fig. 6, only the branching part should be concatenated several times (depending on the allowed communication frequency).

In order to examine the benefits of the information exchange during the cooperative execution, the authors of [10] did not reduce the stopping criterion, allowing all processors to work until the original stopping criterion was satisfied.

The communication points were determined in two different ways: fixed and processor dependent. In the first case, the best solution was exchanged 10 times during the parallel BCO execution regardless the number of processors engaged. In such a way processors were given more freedom to perform independent part of the search. Actually, as the number of processors q increased, the total number of iterations performed before the communication was initiated also increased. On the other hand, the authors of [10] tested how the increasing communication frequency with adding new processors influenced the search process. The idea was to enable passing the information about the improvement of the current global best solution to all the processors as soon as possible. The current global best solution represents the reference point for constructing new solutions and guaranties faster convergence of the resulting search. For the definition of communication points in this case authors [10] used the following rule: current global best solution was exchanged each $n_{it}/(10 * q)$ iterations where n_{it} represented the maximum allowed number of iterations.

5.3.3. Asynchronous cooperation of BCO algorithms

To decrease the communication and synchronization overhead during the cooperative execution of different BCO algorithms, the authors of [10] proposed the use of the asynchronous execution strategy. They implemented this strategy in two different ways, but under the common name *General BCO* (GBCO). The first implementation involved a centrally coordinated knowledge exchange, while the second utilized non-centralized parallelism. Each processor executed a particular sequential variant of BCO until some predefined communication condition was satisfied. It than informed others about its own search status, collected the current global best, and continued its execution. As the asynchronous concept, this strategy does not require all processors to participate in the communication at the same time. Each processor can individually decide when to send its results and when to collect the results arrived from the others. Non-blocking message passing interface and the large enough mailbox buffer were used to support the implementation of this strategy.

The first asynchronous approach proposed in [10] assumed the existence of a central blackboard (a kind of global memory) - to which each processor has access. The communication condition was defined as *the improvement of the current best solution or the execution of 5 iterations without improvement*.

The stopping criterion was not reduced and was set to maximum allowed CPU time in order to ensure better load balancing, i.e., all processors would complete their execution at approximately the same time.

Non-centralized asynchronous parallel BCO execution assumed the existence of several blackboards so that only a subset of (adjacent) processors may post and access information on the corresponding blackboard. In this case, each processor was allowed to perform a single iteration of the corresponding BCO before addressing its associated blackboard. In case that it managed to improve the current best solution from the blackboard, it would post that information on the blackboard and check if there were better solutions already posted there. The best

posted solution would be adopted as the new reference point. If the improvement did not occur in the current iteration, the corresponding processor would simply check for a better solution on its associated blackboard. If a better solution was posted, it would serve as a new reference point, otherwise, the execution would continue with the previous best as the reference point.

5.4. The Artificial Bees and Fuzzy Logic

In the majority of the real life problem models it is assumed that problem data (costs, capacity, distance, duration etc.) are deterministic quantities known in advance. On the other hand, the travel time between two nodes in a network, for example, may involve an uncertainty due to traffic conditions, type of driving, weather conditions, choice of streets, and so on. The most common way to model uncertainty is to use fuzzy logic [60]. Most sets in reality have no sharp line to distinguish between the elements in the set and those outside it. The simplest examples of fuzzy sets are classes of elements characterized by adjectives: big, small, fast, old, etc. The membership function is associated with fuzzy sets and it takes continuous values from the closed interval $[0, 1]$.

In the fuzzy set theory, each set $\mathcal{A}$ is defined as a set of ordered pairs, namely $\mathcal{A} = \{x, \mu_A(x)\}$ where $\mu_A(x)$ indicates the grade of membership for an element x to the set $\mathcal{A}$ [60].

For example, if x is the travel time between nodes i and j, then short could be considered as a particular value of the fuzzy variable travel time x. To each x, a number $\mu_A(x) \in [0, 1]$ is assigned showing the extent to which x is considered to be short. The fuzzy set of subjectively estimated travel time between nodes i and j is denoted by $\mathcal{T}$. In order to simplify the arithmetic operations, the travel time $\mathcal{T}$ is assumed to be a triangular fuzzy number. The triangular fuzzy number $\mathcal{T}$ is expressed as:

$$\mathcal{T} = (t_1, t_2, t_3) \tag{5}$$

where t_1, t_2, and t_3 are the lower boundary, the value that corresponds to the highest grade of the membership, and the upper boundary of fuzzy number $\mathcal{T}$, respectively [25].

The BCO algorithm application on models with fuzzy logic is almost the same as application on deterministic models. Main differences are in the part when:

- bees' partial solutions are compared;
- different component attractiveness is calculated.

In the first case, Kaufmann and Gupta's method can be used to compare fuzzy numbers. In the second case, the approximate reasoning algorithm for calculating the solution component attractiveness could be applied. This algorithm is usually composed from the rules of the following type (Fig. 7):

If the attributes of the solution component are VERY GOOD
Then the considered solution component is VERY ATTRACTIVE

The approximate reasoning based on Fuzzy Logic has been used in [30] to model uncertain demands in nodes when solving vehicle routing problem, and in [50, 51] to model some uncertain quantities for solving Ride-Matching problem.

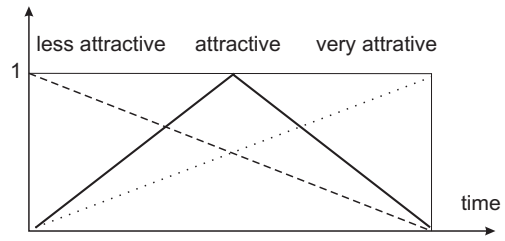

Figure 7: Fuzzy sets describing attractiveness

5.5. BCO and multi-objective programming

The authors of [47] solved the inspection stations location problem by combining the BCO algorithm and compromise programming. This was the first attempt to hybridize BCO with a technique that belongs to the multi-objective programming.

The basic idea behind compromise programming is to create a vector which is called the *ideal vector* and is represented by $[f_1^o, f_2^o, \ldots, f_K^o]$, where f_i^o $(i = 1, 2, \ldots, K)$ denotes the optimum of the i-th objective function. The point that determines the ideal vector is called ideal point. In real-life applications, it is rare, if not impossible, to discover the ideal solution of the considered multi-objective problem. In [47], the following measure of "possible closeness to ideal solution" was proposed.

$$L_p = \left[\sum_{i=1}^{K} w_i^p \left| \frac{f_i(\vec{x}) - f_i^o}{f_{i\,\max} - f_i^o} \right|^p \right]^{1/p} \tag{6}$$

where

$f_i(\vec{x})$ - i-th objective function value that is a result of implementing decision $\vec{x}$;

f_i^o - the optimum value of the i-th objective function;

$f_{i\,\max}$ - the worst value obtainable for the i-th objective function;

K - total number of objective functions;

w_i - i-th objective function's weight;

p - the value that shows distance type: for $p = 1$, all deviations from optimal solutions are in direct proportion to their size, while $2 \leq p \leq \infty$, bigger deviation carry larger weight in L_p metric.

Various compromise solutions can be generated by choosing different values for the parameters. In this way, several feasible alternatives can be presented to the decision-maker.

The authors of [47] assumed that decisions related to the inspection station locations depend on two conflicting objective functions. The first was the minimization of the total number of deployed inspection facilities, and the second was the maximization of the risk reduction. The BCO algorithm for the multi-objective problem formulation, presented in [47], is similar to the BCO application in the

single objective problem case. The difference is only in the part when bees compare their partial solutions. In the multi-objective case, the authors in [47] used L_p metric to perform partial solutions comparison in the following way.

$$L_p = \sqrt{w_1^2 \left| \frac{NF_b - ONF_b}{WNF_b - ONF_b} \right|^2 + w_2^2 \left| \frac{RR_b - ORR_b}{WRR_b - ORR_b} \right|^2}, \quad b = 1, 2, \ldots, B \tag{7}$$

The following labels are used in the previous formula (7):
w_1 - weight of the first criterion;
w_2 - weight of the second criterion;
NF - the number of deployed inspection facilities in the current partial solution;
ONF - the optimal value for the first criterion;
WNF - the worst value for the first criterion;
RR - the value of risk reduction in the current partial solution;
ORR - the optimal value for the second criterion;
WRR - the worst value for the second criterion.

In [47], it was adopted that the worst value obtainable for the first criterion was equal to m (the maximum number of inspection station that could be deployed in the network). The ideal (optimal) value in the case of the first criterion was equal to zero. On the other hand, the worst value for risk reduction was zero. The ideal (optimal) value was equal to the sum of all a_{ip}, where a_{ip} denoted the reduction of risk achieved on path p if the first facility encountered along that path was located at vertex i.

6. THE THEORETICAL VERIFICATION OF THE BCO ALGORITHM

Each implementation of the BCO algorithm is tailored for a particular optimization problem that is to be solved. This involves the solution representation, the rules for constructing/modifying current solutions, and the evaluation and comparison of these solutions. Once the program is completed, it is executed on various instances until the stopping criterion is satisfied. For the considerations to follow, we assume that the stopping criterion is defined as the maximum number of iterations.

The final solution of the BCO execution is identified as the best solution found before the stopping criterion is fulfilled. If the optimal solution for a particular problem instance is not known, we cannot discuss the quality of the final BCO solution. Is it the optimal one or, if not, how far is it from the desired optimum? The only thing we can do is to increase the maximum number of iterations and, possibly, obtain a better final solution.

Numerous successful applications of the BCO method have illustrated its efficiency in an empirical way. Moreover, there are some recent papers dealing with the empirical evaluation [36] and parameter calibration [31] of BCO. However, the theoretical explanations of its effectiveness appeared just recently [20, 21]. In [20], the necessary conditions assuring that an optimal solution can be generated by any bee when the number of iterations is sufficiently large were identified.

Then, the so-called *best-so-far* convergence of the BCO algorithm was proven. It was shown that the current best solution converges to one of the optimal solutions, as the number of iterations increases, with the probability one. This type of convergence is quite common and holds even for some simple, single solution based, stochastic search techniques, like e.g., *random walk*.

Here, we reproduce the first theorem from [20] related to the earlier versions of BCO, which did not include the global knowledge.

Theorem 6.1. *[20]: Let $P^*(t)$ be the probability that the BCO algorithm generates an optimal solution s^* at least once within the first t iterations. Let p^* be the probability that any bee generates the optimal solution. Then for an arbitrary $\varepsilon > 0$ and for a sufficiently large t, $p^* > 0$, implies $P^*(t) \geq 1 - \varepsilon$. Asymptotically, this yields*

$$\lim_{t \to \infty} P^*(t) = 1.$$

Generalization to the more sophisticated case, involving the dependency and knowledge exchange between the iterations, is straightforward. In this case, p^* changes from iteration to iterations and we denote it by $p^*(t)$. Now, only additional requirement is the existence of the lower bound p_{min} on the probability $p^*(t)$, and the same proposition holds referring to p_{min} instead of p^*.

Therefore, the necessary condition for an implementation of the BCO algorithm to be convergent in the best so far sense is that selected solution representation and construction/modification rules allow generating an optimal solution with the probability larger than zero. In another words, the BCO implementation has to ensure that the entire solution space will be visited if the algorithm is given enough time. This means that at least one bee will generate an optimal solution for a given problem. For example, in solving TSP if the solution is represented as a permutation of cities, than the probability to generate an optimal tour equals at least $1/n! > 0$. In solving the p-median or p-center problem, to obtain the optimal solution, a bee has to select appropriate p among n locations. The number of all selections of p elements out of the given n is $\binom{n}{p}$ and therefore, $p^* \geq 1/\binom{n}{p} > 0$.

The results presented in [21] are dealing with different types of convergence as defined in [19] for a general meta–heuristic concept. In addition to the best-so-far convergence of BCO, so called *model convergence* was also proved for the constructive variant of the BCO algorithm. Model convergence assumes learning from the previous experience and therefore, can be considered only for the variants with global knowledge exchange between iterations. Moreover, iteration-dependent probabilities for selecting components during the forward pass (step 2 (a)(i)) have to satisfy some additional properties summarized in Theorem 2.

Theorem 6.2. *[21] Assume that*

$$1 \geq \lambda_t \geq \frac{\log t}{\log(t+1)} \text{ for all } t \geq t_0, \ (t_0 \geq 1)$$

and

$$\sum_{t=1}^{+\infty} (1 - \lambda_t) = +\infty.$$

Then the corresponding BCO algorithm converges in probability to one of the optimal solutions.

Here, λ_t represents the time dependent learning rate used to modify the selection probability of component i in the iteration t ($p_i(t)$). This probability is calculated as follows:

$$p_i(t) = \begin{cases} 1-\lambda_t \cdot (1-p_i(t-1)) & \text{if } i \in x_{t-1}^{bsf}; \\ \lambda_t \cdot p_i(t-1) & \text{if } i \notin x_{t-1}^{bsf}; \\ p_i(1) & \text{if } i \text{ was not choosen.} \end{cases}$$

where $p_i(1)$ denotes the initial (problem specific) probability to choose a component i. The idea is to learn the influence of each component to the quality of generated solution from the previous experience. If the component i is part of the current best (best-so-far) solution, the probability that it will be selected in the next iteration is increased. If this component was included in some low quality solutions, the probability of its selection for the next iteration is decreased. Obviously, if a component was not included in any solution, we cannot evaluate its influence. In that case, its probability remains unchanged in order to give it more chance to be chosen in the iterations to follow. As the number of iterations increases, the components not included in the search will have larger selection probability then those belonging to the low quality solutions and therefore, their chances to be included in some future solutions will increase. Consecutively, as the number of iterations $t \to +\infty$, only first two modification cases will remain. Complete explanations and proof of Theorem 2 are beyond the scope of this paper; interested readers are referred to [21].

7. CONCLUSION

The Bee Colony Optimization algorithm, one of the Swarm Intelligence techniques, is a meta–heuristic method inspired by the foraging behavior of honeybees. It represents a general algorithmic framework applicable to various optimization problems in management, engineering, control, etc., and should always be *tailored* for a specific problem. The BCO method is based on the concept of *cooperation*, which increases the efficiency of artificial bees. BCO has the capability to intensify search in the promising regions of the solution space through information exchange and recruiting process. The diversification process is realized by restricting the search within different iterations.

This paper contains a brief survey of all algorithms based on foraging habits of honeybees, the detailed explanation related to the BCO algorithm proposed in [27, 28, 29, 30], and all modifications applied on initial version of the algorithm during its various applications. Moreover, it briefly describes recently obtained mathematical validation of the BCO approach. Detailed survey of BCO applications to various combinatorial and continuous optimization problems is presented in our paper: Bee Colony Optimization Part II: The Application Survey.

In years to come, the authors expect more BCO based models, examining, for instance, bees' homogeneity (homogenous vs. heterogeneous artificial bees), various information sharing mechanisms, and various collaboration mechanisms.

Acknowledgements. This work has been partially supported by the Serbian Ministry of Education Science and Technological Development, grants No. OI174010, OI174033, and TR36002. The authors would like to express gratefulness to the anonymous referees for their valuable comments and suggestions that helped to improve the presentation of the paper.

REFERENCES

[1] Abbass, H. A., "MBO: Marriage in honey bees optimization-a haplometrosis polygynous swarming approach", *Proceedings of the Congress on Evolutionary Computation*, Seoul, South Korea, 1 (2001) 207–214..

[2] Afshar, A., Bozorg Haddad, O., Mariño, M.A., and Adams B.J., "Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation", *J. Franklin Institute*, 344(5) (2007) 452–462.

[3] Anantasate, S., Chokpanyasuwan, C., Pattaraprakorn, W., and Bhasaputra, P., "Multi-objective optimal placement of distributed generation using bee colony optimization", *GMSARN International Journal*, 3 (2009) 55–64.

[4] Banharnsakun, A., Achalakul, T., and Sirinaovakul, B., "Artificial bee colony algorithm on distributed environments", *Proc. Second World Congress on Nature and Biologically Inspired Computing (NaBIC'10)*, Fukuoka, 1 (2010) 13–18.

[5] Beni, G., and Wang, J., "Swarm intelligence", *Proc. Seventh Annual Meeting of the Robotics Society of Japan*, RSJ Press, Tokyo, 1 (1989) 425–428.

[6] Bonabeau, E., Dorigo, M., and Theraulaz, G., *Swarm Intelligence*, Oxford University Press, Oxford, 1997.

[7] Camazine, S., and Sneyd, J., "A model of collective nectar source by honey bees: self-organization through simple rules", *Journal of Theoretical Biology*, 149 (1991) 547–571.

[8] Chong, C. S., Low, M. Y. H., Sivakumar, A. I., and Gay, K. L., "A bee colony optimization algorithm to job shop scheduling", In *Proceedings of the Winter Simulation Conference*, Washington, DC, 1 (2006) 1954–1961.

[9] Crainic, T. G., Davidović, T., and Ramljak, D., "Designing parallel meta-heuristic methods", In M. Despotović-Zrakić, V. Milutinović, and A. Belić, editors, *High Performance and Cloud Computing in Science and Education*, IGI-Global, (2014) 260–280.

[10] Davidović, T., Jakšić, T., Ramljak, D., Šelmić, M., and Teodorović, D., "MPI parallelization strategies for bee colony optimization", *Optimization, Special Issue entitled "Advances in Discrete Optimization" dedicated to BALCOR 2011*, 62(8) (2013) 1113–1142.

[11] Davidović, T., Ramljak, D., Šelmić, M., and Teodorović, D., "MPI parallelization of bee colony optimization", *Proc. 1st International Symposium & 10th Balkan Conference on Operational Research*, Thessaloniki, Greece, 2 (2011) 193–200.

[12] Davidović, T., Ramljak, D., Šelmić, M., and Teodorović, D., "Bee colony optimization for the *p*-center problem", *Computers and Operations Research*, 38(10) (2011) 1367–1376.

[13] Davidović, T., Šelmić, M., and Teodorović, D., "Scheduling independent tasks: Bee colony optimization approach", *Proc. 17th Mediterranean Conference on Control and Automation*, Makedonia Palace, Thessaloniki, Greece, 1 (2009) 1020–1025.

[14] Davidović, T., Šelmić, M., Teodorović, D., and Ramljak, D., "Bee colony optimization for scheduling independent tasks to identical processors", *J. Heur.*, 18(4) (2012) 549–569.

[15] Dimitrijević, B., Teodorović, D., Simić, V., and Šelmić, M., "Bee colony optimization approach to solving the anticovering location problem", *Journal of Computing in Civil Engineering*, 26(6) (2011) 759–768.

[16] Drias, H., Sadeg, S., and Yahi, S., "Cooperative bees swarm for solving the maximum weighted satisfiability problem", *LNCS: Computational Intelligence and Bioinspired Systems*, 3512 (2005) 318–325.

[17] Edara, P., Šelmić, M., and Teodorović, D., "Heuristic solution algorithms for a traffic sensor optimization problem", *INFORMS 2008*, Washington D.C., Oct. 12–15, (2008).
[18] Fonseca, R., Paluszewski, M., and Winter, P., "Protein structure prediction using bee colony optimization metaheuristic", *J. Math. Model. Alg.*, 9(2) (2010) 181–194.
[19] Gutjahr, W. J., "Convergence analysis of metaheuristics", In V. Maniezzo, T. Stèutzle, and S. Voss, editors, *Matheuristics: hybridizing metaheuristics and mathematical programming*, Springer, 10 (2009) 159–187.
[20] Jakšić Krüger, T., "On the convergence of the bee colony optimization meta-heuristic", *Proc. Fourth Symposium "Mathematics and Applications"*, Faculty of Mathematics, University of Belgrade, May, 24–25, Serbia (in Serbian), IV(1) (2013) 176–187.
[21] Jakšić Krüger, T., Davidović, T., Teodorović, D., and Šelmić, M., "The bee colony optimization algorithm and its convergence", *International Journal of Bio-Inspired Computation*, (accepted), (2014).
[22] Karaboga, D., "An idea based on honey bee swarm for numerical optimization", Technical report, Erciyes University, Engineering Faculty Computer Engineering Department Kayseri/Turkiye, (2005).
[23] Karaboga, D., and Akay, B., "A survey: Algorithms simulating bee swarm intelligence", *Artificial Intelligence Review*, 31 (2009) 61–85.
[24] Karaboga, D., Gorkemli, B., Ozturk, C., and Karaboga, N., "A comprehensive survey: Artificial bee colony (ABC) algorithm and applications", *Artificial Intelligence Review*, 42(1) (2014) 21–57.
[25] Kaufmann, A., and Gupta, M., *Introduction to Fuzzy Arithmetic*, New York: Van Nostrand Reinhold Company, (1988).
[26] Kovač, N., "Bee colony optimization algorithm for the minimum cost berth allocation problem", *Proc. XI Balkan Conference on Operational Research*, (BALCOR, 2013), Sept. 07–11, Beograd–Zlatibor, Serbia, 1 (2013) 245-254.
[27] Lučić, P., and Teodorović, D., "Bee system: Modeling combinatorial optimization transportation engineering problems by swarm intelligence", *Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis*, Sao Miguel, Azores Islands, (2001) 441–445.
[28] Lučić, P., and Teodorović, D., "Transportation modeling: An artificial life approach", *Proc. 14th IEEE International Conference on Tools with Artificial Intelligence*, Washington, DC, (2002) 216–223.
[29] Lučić, P., and Teodorović, D., "Computing with bees: Attacking complex transportation engineering problems", *International Journal on Artificial Intelligence Tools*, 12(3) (2003) 375–394.
[30] Lučić, P., and Teodorović, D., "Vehicle routing problem with uncertain demand at nodes: The bee system and fuzzy logic approach", In J. L. Verdegay, editor, *Fuzzy Sets based Heuristics for Optimization*, Physica Verlag: Berlin Heidelberg, (2003) 67–82.
[31] Maksimović, P., and Davidović, T., "Parameter calibration in the bee colony optimization algorithm", *Proc. XI Balcan Conference on Operational Research*, (BALCOR 2013), Sept. 07–11, Beograd-Zlatibor, Serbia, 1 (2013) 263–272.
[32] Marković, G., Teodorović, D., and Aćimović-Raspopović, V., "Routing and wavelength assignment in all-optical networks based on the bee colony optimization", *AI Commun.*, 20(4) (2007) 273–285.
[33] McFadden, D., "Conditional logit analysis of quantitative choice behavior", In P. Zaremmbka, editor, *Frontier of Econometrics*, Academic Press, New York, (1973).
[34] Nakrani, S., and Tovey, C., "On honey bees and dynamic server allocation in internet hosting centers", *Adapt. Syst.*, 12(3–4) (2004) 223–240.
[35] Narasimhan, H., "Parallel artificial bee colony (PABC) algorithm", *Proc. VIII International Conference on Computer Information Systems and Industrial Management (CISIM, 2009)*, World Congress on Nature and Biologically Inspired Computing (NaBIC'09), Coimbatore, (2009).
[36] Nikolić, M., and Teodorović, D., "Empirical study of the bee colony optimization (BCO) algorithm", *Expert Systems with Applications*, 40(11) (2013) 4609–4620.
[37] Nikolić, M., and Teodorović, D., "Transit network design by bee colony optimization", *Expert Systems with Applications*, 40(15) (2013) 5945–5955.
[38] Parpinelli, R. S., Benitez, C. M. V., and Lopes, H. S., "Parallel approaches for the artificial bee colony algorithm", In B. K. Panigrahi, Y. Shi, and M-H. Lim, editors, *Handbook of Swarm Intelligence: Concepts, Principles and Applications*, volume Series: Adaptation, Learning, and Optimization, Springer, Berlin, Germany, 8 (2011), 329–346.

[39] Pham, D. T., Ghanbarzadeh, A., Koc, E., Otri, S., and Zaidi, M., "The bees algorithm - a novel tool for complex optimisation problems", *Proc. 2nd Virtual International Conference on Intelligent Production Machines and Systems (IPROMS 2006)*, Elsevier, Cardiff, Wales, UK, (2006) 454–459.
[40] Pham, D. T., Soroka, A. J., Ghanbarzadeh, A., and Koc, E., "Optimising neural networks for identification of wood defects using the bees algorithm", *Proc. IEEE International Conference on Industrial Informatics*, Singapore, (2006) 1346–1351.
[41] Sato, T., and Hagiwara, M., "Bee system: Finding solution by a concentrated search", *Proc. IEEE International Conference on Systems, Man, and Cybernetics 'Computational Cybernetics and Simulation'*, Orlando, FL, USA, (1997) 3954–3959.
[42] Seeley, T., *Honeybee Ecology: A Study of Adaptation in Social Life*. Princeton Univ. Press, (1995).
[43] Srichandum, S., and Rujirayanyong, T., "Production scheduling for dispatching ready mixed concrete trucks using bee colony optimization", *Amer. J. Engineer. App. Sci.*, 3(1) (2010) 7–14.
[44] Stojanović, T., Davidović, T., and Ognjanović, Z., "Bee-colony optimization for the satisfiability problem in probabilistic logic", *(submitted for publication)*, (2013).
[45] Subotić, M., Tuba, M., and Stanarević, N., "Different approaches in parallelization of the artificial bee colony algorithm", *International Journal Of Mathematical Models And Methods in Applied Sciences*, 5(4) (2011) 755–762.
[46] Suguna, N., and Thanushkodi, K., "A novel rough set reduct algorithm for medial domain based on bee colony optimization", *J. Comput.*, 2(6) (2010) 49–54.
[47] Šelmić, M., Teodorović, D., and Vukadinović, K., "Locating inspection facilities in traffic networks: An artificial intelligence approach", *Transport. Plan. Techn.*, 33(6) (2010) 481–493.
[48] Teodorović, D., "Swarm intelligence systems for transportation engineering: Principles and applications", *Transp. Res. Pt. C-Emerg. Technol.*, 16 (2008) 651–782.
[49] Teodorović, D., "Bee colony optimization (BCO)", In C. P. Lim, L. C. Jain, and S. Dehuri, editors, *Innovations in Swarm Intelligence*, Springer-Verlag, Berlin Heidelberg, (2009) 39–60.
[50] Teodorović, D., and Dell'Orco, M., "Bee colony optimization - a cooperative learning approach to complex transportation problems", In *Advanced OR and AI Methods in Transportation. Proceedings of the 10th Meeting of the EURO Working Group on Transportation*, Poznan, Poland, (2005) 51–60.
[51] Teodorović, D., and Dell'Orco, M., "Mitigating traffic congestion: Solving the ride-matching problem by bee colony optimization", *Transport. Plan. Techn.*, 31(2) (2008) 135–152.
[52] Teodorović, D., Šelmić, M., and Mijatović-Teodorović, Lj., "Combining case–based reasoning with bee colony optimization for dose planning in well differentiated thyroid cancer treatment", *Expert Systems with Applications*, 40(6) (2013) 2147–2155.
[53] Teodorović, D., and Šelmić, M., "The BCO algorithm for the *p*-median problem", *Proc. XXXIV Serbian Operations Research Conferece*, Zlatibor, Serbia (in Serbian), (2007).
[54] Vassiliadis, V., and Dounias, G., "Nature inspired intelligence for the constrained portfolio optimization problem", In *Artificial Intelligence: Theories, Models and Applications, LNCS, vol. 5138*, Springer-Verlag, Berlin–Heidelberg, (2008) 431–436.
[55] Wedde, H. F., Farooq, M., and Zhang, Y., "BeeHive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior", *LNCS: Ant Colony Optimization and Swarm Intelligence*, 3172 (2004) 83–94.
[56] Wong, L.-P., Hean Low, M. Y., and Chong, C. S., "A bee colony optimization algorithm for traveling salesman problem", *Proc. 2-nd Asia International Conference on Modelling & Simulation*, Kuala Lumpur, Malaysia, (2008) 818–823.
[57] Wong, L.-P., Hean Low, M. Y., and Chong, C. S., "An efficient bee colony optimization algorithm for traveling salesman problem using frequency-based pruning", *Proc. 7-th IEEE International Conference on Industrial Informatics*, Cardiff, Wales, UK, June 24–26, (2009) 775–782.
[58] Yang, C., Chen, J., and Tu, X., "Algorithm of fast marriage in honey bees optimization and convergence analysis", *Proc. IEEE International Conference on Automation and Logistics*, Jinan, China, (2007) 1794–1799.
[59] Yang, X-S., "Engineering optimizations via nature-inspired virtual bee algorithms", In J. Mira and J. R. Alvarez, editors, *Proc. IWINAC International Work-conference on the Interplay between Natural and Artificial Computation*, IWINAC 2005, LNCS 3562, Las Palmas de Gran Canaria, Canary Islands, Spain, June 15–18, (2005) 317–323.
[60] Zadeh, L., "Fuzzy sets", *Information and Control*, 8 (1965) 338-353.