Yugoslav Journal of Operations Research
25 (2015), Number 2, 185–219
DOI: 10.2298/YJOR131029020T

Invited survey

BEE COLONY OPTIMIZATION PART II: THE APPLICATION SURVEY

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

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

Received: October 2013 / Accepted: November 2013

Abstract: Bee Colony Optimization (BCO) is a meta–heuristic method based on foraging habits of honeybees. This technique was motivated by the analogy found between the natural behavior of bees searching for food and the behavior of optimization algorithms searching for an optimum in combinatorial optimization problems. BCO has been successfully applied to various hard combinatorial optimization problems, mostly in transportation, location and scheduling fields. There are some applications in the continuous optimization field that have appeared recently. The main purpose of this paper is to introduce the scientific community more closely with BCO by summarizing its existing successful applications.

Keywords: Meta–Heuristic Methods, Swarm Intelligence, Combinatorial Optimization, Routing, Location, Scheduling Problems.

MSC: 90-03, 68W20, 90BXX.

1. INTRODUCTION

The Bee Colony Optimization (BCO) meta-heuristic, inspired by foraging behavior of honeybees, was proposed in [18, 19, 20, 21] for dealing with the well

known hard combinatorial optimization problems: travelling salesman and vehicle routing. The plan was to build the multi agent system (a colony of artificial bees) able to efficiently solve hard optimization problems.

BCO is a stochastic, random-search population-based technique. It was motivated by the analogy found between the natural behavior of bees searching for food and the behavior of optimization algorithms searching for an optimum in combinatorial optimization problems. Artificial bees investigate through the search space looking for feasible solutions. In order to increase the quality of produced solutions, autonomous artificial bees collaborate and exchange information. Sharing the available information and using collective knowledge, artificial bees concentrate on more promising areas, and slowly abandon solutions from those less promising. Step by step, artificial bees collectively generate and/or improve their solutions. BCO performs its search in iterations until some predefined stopping criterion is satisfied.

The BCO meta-heuristic has been recently used as a tool for treating large and complex real-world problems. It has been shown that BCO possesses an ability to find high quality solutions to difficult combinatorial problems within a reasonable amount of running time. This paper presents the classification and analysis of the results achieved using BCO to model complex science and engineering optimization problems in the past decade. First, we shortly describe the BCO algorithm (detailed explanation of the algorithm steps, evolution and modification of BCO can be found in the previous paper, Bee Colony Optimization Part I: The Algorithm Overview [5]). Later on, we describe all BCO applications that we are aware of. BCO has been successfully applied to various problems by Teodorović and co-authors [3, 6, 7, 8, 18, 19, 20, 21, 22, 28, 29, 40, 41, 36, 37, 39, 38]. Recently, some other researchers also used BCO meta–heuristic as a tool to solve numerous, complex problems [15, 16, 26, 27, 31, 32, 33, 42, 44, 45].

The paper is organized as follows. After this introduction, brief description of the BCO algorithm is given in Section 2. Section 3 contains survey of BCO applications divided into six groups by the type of problems: routing, networks, location, scheduling, medicine with chemistry and continuous optimization problems. The last section contains some concluding remarks.

2. BRIEF DESCRIPTION OF THE BCO ALGORITHM

Lučić and Teodorović [18, 19, 20, 21] were among the first who used basic principles of collective bee intelligence in solving combinatorial optimization problems. BCO is a population based algorithm: population of B artificial bees searches for the optimal solution of a given optimization problem. Every artificial bee generates one solution to the problem. The algorithm consists of two alternating phases: *forward pass* and *backward pass*. During each forward pass, all bees are exploring the search space by applying a predefined number of moves, which construct and/or improve the solution, yielding a new solution.

Having obtained new partial/complete solutions, the bees start executing a second phase, the so-called backward pass. During the backward pass, all bees

share information about their solutions. In nature, bees would perform a dancing ritual, which would inform other bees about the amount of food they have found, and the proximity of the patch to the hive. In the search algorithm, the quality of each solution is defined as the current value of the objective function. 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 selection is taken with a probability, such that better advertised solutions have greater opportunities to be chosen for further exploration.

The two phases of the search algorithm (forward and backward pass) alternate NC times, i.e., until each bee completes the generation of its solution or performs NC solution modifications. Parameter NC is 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 can 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, maximum allowed CPU time, etc. At the end, the best found solution (the so called global best) is reported as the final one. The pseudo-code of the BCO algorithm is given in Fig. 1.

Steps (1), (a), and (b) are problem dependent and should be resolved in each particular implementation of the BCO algorithm. On the other hand, there are formulae specifying steps (c), loyalty decision, and (d), recruiting process, and they are described in the paper Bee Colony Optimization Part I: The Algorithm Overview [5], Subsections 4.1 and 4.2.

3. BCO APPLICATIONS

This section summarizes the applications of the above described BCO method or its variations. According to the best of our knowledge, BCO has been applied to the following classes of problems:

- Routing: the traveling salesman problem [18, 43, 44, 45], vehicle routing problem [21], and the routing and wavelength assignment (RWA) in all-optical networks [22].
- Location: the p-median problem [39], traffic sensors locations problem on highways [40], inspection stations locations in transport networks [41], anti-covering location problem [8], p-center problem [3] location of distributed generation resources [33], and capacitated plant location problem [16].
- Scheduling: static scheduling of independent tasks on homogeneous multiprocessor systems [6, 7], the ride-matching problem [36, 37], job shop

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 1: Pseudo-code for BCO

scheduling [31], task scheduling in computational grids [26], backup allocation problem [27], and berth allocation problem [15].

- Medicine with chemistry: cancer therapy [38]; chemical process optimization [32].
- Networks: network design [29].
- Continuous and mixed optimization problems: numerical function minimization [28]; the satisfiability problem in probabilistic logic [34].

These applications are explained in some more details in the rest of this section.

3.1. Application of BCO to Routing Problems

3.1.1. Solving the Traveling Salesman Problem by BCO

In [18, 19, 20], the authors tested the early version of the BCO approach on the well known Traveling Salesman Problem (TSP). TSP is defined in the following way: Given n nodes, find the shortest itinerary that starts in a specific node, goes through all other nodes exactly once and finishes in the starting node.

In the approach proposed in [18, 19, 20], the random selection of an initial node was represented by changing the location of a hive at the beginning of each iteration. The TSP problem was decomposed into stages. At each stage

(corresponding to the forward pass of BCO), a bee chooses the new nodes to be added to the partial Traveling Salesman tour created so far. This selection was performed in probabilistic random manner. The authors proposed Logit-based model [23] for calculating the probability of choosing next node to be visited. Logit model is one of the most successful and widely accepted discrete choice models. For calculating the probability, the model proposed in [18, 19, 20] took into account the following data: the distance between current node (hive at the beginning) and the node-candidate to be visited; the total number of performed iterations in a search process, and the total number of bees that have visited the considered link in the past. The proposed model used complex and complicated formulae, and was not used in subsequent implementations or by other researchers.

During the backward pass, each bee decided whether to keep and advertize the generated partial solution or to abandon it (i.e., to return to its role of an uncommitted follower). The choice was made with a certain probability in such a way that bees with better partial solutions had greater chance to continue their own exploration. Each follower had chosen a new solution among the recruiters by the roulette wheel, where better solutions had higher probability of being chosen for further exploration. After the selection had been made, bees expanded previously generated partial solutions by a predefined number of nodes during the next forward pass, followed by the corresponding backward pass. These steps (forward/backward passes) were repeated until complete solutions were generated (for each bee the whole TSP tour was discovered). The authors of [20] tried to improve the obtained solutions by applying different tour improvement algorithms based on k-opt procedure at the end of each iteration. Finally, among all generated solutions, the best one was determined and used to update the current global best. This represented the end of a single iteration, and the next one started after the hive relocation. The effectiveness of this early version of BCO was tested on a large number of numerical examples for benchmark problems taken from OR-Library.

TSP was also considered in [43]. In that BCO algorithm bees were used to generate feasible solutions for TSP benchmark problems. These solutions were then improved by a local search. The main distinctions between model proposed in [43] and the original one from [18] are: (1) Bees do not have the ability to remember the search history expressed by a number of bees that have already visited an arc; (2) Bees advertize the entire feasible path rather than partial solution during the dance ritual; (3) The hive position is set prior to the algorithm execution to a virtual position on equal distance to all other cities; (4) Bees are influenced by both arc fitness and distance between nodes when constructing the solutions.

The algorithm was tested on the large number of benchmark problems and it managed to improve some of the results from [18]. To improve previously obtained results, a new model was proposed in [44]. The authors implemented two mechanisms for 2-opt local search, named frequency-based pruning strategy (FBPS) and fixed-radius near neighbor (FRNN). The first strategy, FBPS assumed that only a subset of promising solutions was allowed to perform 2-opt local search based on the accumulated frequency of its building blocks. FRNN 2-

opt was declared as an efficient implementation of 2-opt which exploited the geometric structure in a permutation of TSP sequence. The experimental results showed that they were able to achieve a 58.42% improvement with respect to the original BCO algorithm while maintaining the solution quality at 0.02% from the known optimal. The extended study involving all BCO variants was presented in [45]. Experimental results comparing the proposed BCO algorithm with existing approaches on a set of benchmark problems were presented. For 84 benchmark problems, the BCO algorithm was able to obtain an overall average solution deviation of 0.31% from the known optimum. The results also showed that BCO was comparable with other algorithms such as Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO).

3.1.2. BCO for Routing and Wavelength Assignment (RWA) in All-Optical Networks

The Routing and Wavelength Assignment (RWA) in All-Optical Networks is the well-known optimization problem in telecommunication. The RWA problem could be described in the following way: Assign a path through the network and a wave-length on that path for each considered connection between a pair of nodes in such a way to maximize the total number of established connections in the network. In [22] this problem was addressed by the BCO meta-heuristic. The authors named the proposed algorithm BCO-RWA. An artificial network had been created with nodes representing the collection of all considered origin-destination pairs (Fig. 2). Each artificial node was comprised of an origin and destination linked by a number of routes. Light-path was a route chosen by an artificial bee. The solution generated by each bee during a single flight contained the collection of established light-paths.

During the forward pass, each bee visited n nodes (tried to establish n new light-paths), where n was selected in such a way that $n \ll m$, m representing the total number of requested light-paths. At each node, a bee was choosing among remaining artificial nodes (not previously selected ones). Sequence of n visited artificial nodes, generated by a bee, represented one partial solution of the problem considered. Bee was not always successful in establishing light-path when visiting the artificial node. Bee's success depended on the wavelengths' availability along the specific links. In this way, generated partial solutions differed among themselves according to the total number of established light-paths.

The probability p that a specific unvisited artificial node would be chosen by the bee was set to $1/n_{unvis}$, where n_{unvis} denoted the total number of unvisited artificial nodes. By visiting specific artificial node in the network, bees attempted to establish the requested light-path between one real source-destination node pair in optical network. Under the assumption that a specific bee decided to consider the light-path request between the source node s and the destination node d, it was necessary to choose the route and to assign an available wavelength along the route between these two real nodes. For every node pair (s, d), the authors defined a subset R^{sd} of allowed routes that could be used when establishing the light-path. These subsets were defined by using the k shortest path algorithm: For each of the k possible routes, the utility when bee is choosing the considered route

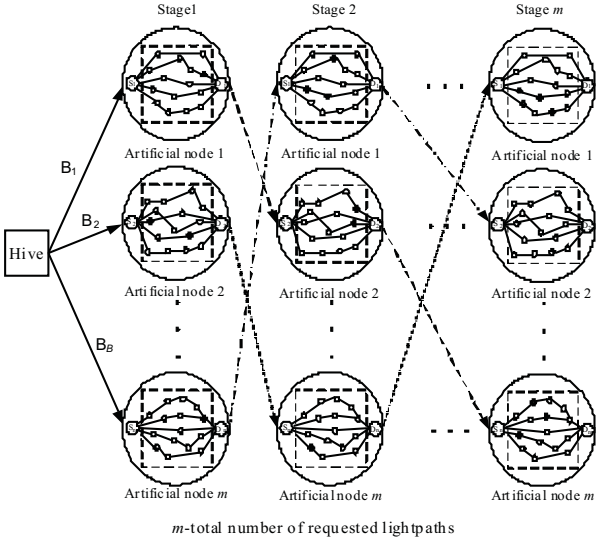

Figure 2: RWA Artificial network

was calculated. The route length and the number of available wavelengths along the route influenced the bee's utilities value: shorter routes with larger number of wavelengths yielded higher utility value. The authors defined the utilities V_r^{sd} of choosing the route r between the node pair (s,d) in the following way:

$$V_r^{sd} = a\,\frac{1}{h_r - h_{r\,\min} + 1} + (1-a)\frac{W_r}{W_{\max}} \tag{1}$$

where:

r - the route ordinary number (index) for a node pair (s,d), $r = 1,2,\ldots,k$, $r \in \{R^{sd}\}$;

h_r - the route length expressed in the number of physical hops;

$h_{r\,\min}$ - the length of the shortest route $r_{\min}$;

W_r - the number of available wavelengths along the route r;

$W_{\max} = \max_{r\in\{R^{sd}\}}\{W_r\}$ - the maximum number of available wavelengths among all routes $r \in \{R^{sd}\}$;

a - weight (importance) of the criterion, $0 \le a \le 1$.

The selection of a physical route in optical network was performed in a random manner, inspired by the Logit model [23]. The probability p_r^{sd} of choosing route r in the case of origin-destination pair (s,d) was defined as:

$$p_r^{sd} = \begin{cases} \dfrac{e^{V_r^{sd}}}{\sum_{i=1}^{|R^{sd}|} e^{V_i^{sd}}}, & \forall r \in \{R^{sd}\} \text{ and } W_r > 0; \\ 0, & \forall r \in \{R^{sd}\} \text{ and } W_r = 0. \end{cases} \tag{2}$$

where $|R^{sd}|$ is the total number of available routes between pair of nodes (s, d). The route r is available if there is at least one available wavelength on all links that belong to that route. The total number of established light-paths from the beginning of current iteration was used as a criterion for the b-th bee partial solution evaluation.

The BCO-RWA algorithm from [22] was tested on a few numerical examples with the number of requested light-paths ranging from 28 to 40. The authors formulated the corresponding Integer Linear Program (ILP) to determine optimal solutions for the considered examples. They compared the BCO-RWA results with the optimal solutions. Their conclusion was that the proposed BCO-RWA algorithm has been able to produce optimal or at least near-optimal solutions in a reasonable amount of CPU time.

3.2. Application of BCO to Location Problems

3.2.1. BCO for the p-Median problem

The problem of locating p facilities in order to minimize the average"distance" between facilities and serviced users is known as a p-median problem, and it is encountered when designing networks for distribution centers, locations for schools, post offices, shops, etc. The p-median problem was formulated for the first time in [12, 13]. It is one of the fundamental problems in the area of discrete location analysis.

The p-median problem could be defined in the following way: Among given n possible locations, it is necessary to establish p facilities (medians) on a network in such a way to minimize the sum of all distances from each demand node to its nearest facility. The p-median problem is NP-hard [14] and is usually treated by various heuristic algorithms, and procedures based on meta-heuristic rules [24]. In [39], the BCO approach to p-median problem was proposed.

In the first step, hive has been located in the node where a median should be positioned in the case $p = 1$. An algorithm based on the matrix of shortest distances between nodes and service demands in all nodes was used. In order to calculate the node attractiveness and to choose the next node to be added to the partial solution, the authors used the concept of node's utility. The utility V_i in the case when a bee chooses node i to be the median was calculated as:

$$V_i = rR_i + qA_i \tag{3}$$

where:

R_i - normalized value of the distance from the hive to the i-th node;

A_i - normalized value of the demand in the i-th node;

r, q - weight (importance) of the distance and the demand, respectively.

Normalized values are calculated as follows:

$$R_i = \frac{r_{\max} - r_i}{r_{\max} - r_{\min}},\ R_i \in [0,\ 1]; \quad A_i = \frac{a_i - a_{\min}}{a_{\max} - a_{\min}},\ A_i \in [0,\ 1]; \tag{4}$$

with

r_i - the distance from the hive to the i-th node;
$r_{\min}$, $r_{\max}$ - minimum and maximum among all r_i distances, respectively;
a_i - demand at node i;
$a_{\min}$, $a_{\max}$ - minimum and maximum among all demands, respectively.
The probability p_i of choosing node i as a median was calculated as follows:

$$p_i = \frac{V_i}{\sum_{k=1}^{K} V_k}, \; i = 1, \ldots, n \tag{5}$$

where K denotes the number of available (not previously chosen) nodes.

The quality of each generated partial solution was evaluated by using the total distance travelled by the clients served at the medians chosen so far. The proposed algorithm was tested on various problem instances ranging in size from instances with $n = 20$ and $p = 2, 3$, and 4 up to the instances with $n = 1000$ and $p = 2, 3$, and 5. Smaller test problems were generated by the authors in [39], while the 1000 nodes examples (Koerkel Problem Sets) were taken from the Internet. The authors compared the results obtained by BCO against the known optimal solutions. They concluded that BCO was able to produce "good" solutions in a "reasonable" computation time. Based on a large number of performed tests, the authors showed that increasing the number of bees did not improve the search: the influence on the solution quality was negligible, while the required computational times became longer.

3.2.2. The BCO Approach to Optimize Locations of Traffic Sensors on Highways

The placement of point sensors within a roadway network problem belongs to the field of location theory. Point sensors are deployed on roadways to collect traffic data including volume, occupancy, and speed. The spacing of sensors on freeways has a key impact on the travel time estimates obtained from the reported speeds. There is a tradeoff between sensor spacing and travel time estimate correctness. Transportation agencies are therefore seeking for a method to indicate the most appropriate locations for sensor deployment such that the travel time estimate error is minimized, within the constraints of available capital and maintenance funding.

In [40] the problem of optimal placing of traffic sensors on freeways was studied and the BCO algorithm for this problem was developed. The proposed model tried to minimize the error in travel time estimation, while taking into account the budget constraints. During the forward pass of the BCO algorithm, the Logit model [23] was used for selecting the potential sensor locations. The probability of choosing a node i by any bee was expressed using the Logit model as follows:

$$p_i = \frac{e^{U_i}}{\sum_{r=1}^{n} e^{U_r}}, \; i = 1, \ldots, n \tag{6}$$

where U_i represented the utility of having a sensor at node i. This utility depended on several factors that may affect travel time estimates. To determine the

utilities various factors could be used: the presence of a natural bottleneck at that location (e.g., a lane reduction) leading to the recurring congestions during the peak traffic periods, historical accident likelihoods (to monitor the induced delays by deploying sensors), level of traffic volumes, etc. In [40] it was assumed that all potential sensor locations have equal utilities. Within each forward pass, all bees visited certain number of nodes and created the corresponding partial solutions (by choosing few nodes to become the sensor locations). Each generated partial solution was characterized by the travel time estimation error. As the criterion for comparison of partial solutions, the maximum travel time error was selected.

Tradeoff plots were generated by varying the actual number of sensors (d) from 2 to 20 by the increment of 1. For a given number of sensors, the obtained optimal placement would result in a travel time estimation error for each travel time run. The BCO obtained results were compared against the sensor deployment obtained by the Genetic Algorithm (GA) from [10]. The BCO results were competitive with the results obtained by GA and enabled savings of 30% with respect to the existing deployment at 20 locations.

3.2.3. The BCO Approach for Locating Inspection Facilities in Traffic Networks

Two models to determine the locations of the incapacitated inspection stations in the traffic network were developed and addressed by BCO in [41]. The first model formulation was related to a single-objective optimization problem. The objective function to be maximized was represented as the largest possible risk decrease on the traffic network. The number of facilities was given in advance in the first version of the model and treated as a constraint in the problem formulation. The multi-objective approach to the problem of inspection stations deployment in traffic networks was analyzed by the second model in [41]. The authors started from the assumption that decisions related to the inspection station locations should be made in the presence of trade-offs between two or more conflicting objectives. As the consequence, several conflicting objectives should be optimized at the same time. The objectives considered in [41] were: the maximization of the risk reduction, and the minimization of the total number of deployed inspection station facilities. The BCO algorithm was applied to the proposed models in order to properly organize checkups of truck weight limits, hours and service regulations, vehicle equipment safety, and prevention of drunk driving.

Single-objective BCO model

The authors of [41] decomposed the problem into stages. The first node to be chosen for the inspection station represented the first stage; the second node to be chosen was a part of the second stage, etc. The probability p_i ($i = 1, 2, \ldots, n$) of choosing node i as an inspection station was calculated as:

$$p_i = \frac{U_i}{\sum_{k=1}^{K} U_k}, \; i = 1, \ldots, n \tag{7}$$

where U_i ($i = 1, 2, \ldots, n$) represented the node utilities, and K denoted the number of not previously chosen nodes. All utilities were given equal values, i.e., all

nodes were considered equally interesting for bees to locate inspection stations in them. Each generated partial solution was characterized by the value of risk reduction.

Multi-objective BCO model

The compromise programming, as a tool for solving multi-objective inspection stations location problem was used in [41] in the second model. Duckstein [9] proposed the following measure of "possible closeness to ideal solution":

$$L_p = \left[\sum_{i=1}^{M} w_i^p \left| \frac{f_i(\vec{x}) - f_i^o}{f_{i\,\max} - f_i^o} \right|^p \right]^{1/p} \tag{8}$$

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;
M - 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.

The BCO approach to the multi-objective problem formulation was similar to the single objective case. The only difference appeared in the part related to the partial solutions comparison mechanism. In the multi-objective case, L_p metric was used to compare partial solutions. The authors adopted the following assumptions: the worst value obtainable for the first criterion equals m (the maximum number of inspection stations that is possible to deploy in the network), and the ideal (optimal) value in the case of first criterion equals zero; the worst value for risk reduction equals zero, while the ideal (optimal) value is equal to the sum of *all risk reduction values*. This was the first attempt in relevant literature to combine the BCO meta-heuristic with multi-objective programming.

The quality of the generated partial solution according to the L_p metric was calculated in the following way:

$$L_p = \sqrt{w_1^2 \left| \frac{NF_b - ONF_b}{WNF_b - ONF_b} \right|^2 + w_2^2 \left| \frac{RR_b - ORR_b}{WRR_b - ORR_b} \right|^2}, \quad b = 1, 2, \ldots, B \tag{9}$$

where:

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.

The proposed approaches were tested on a various test problems, divided into three groups. The first two groups (easy and medium examples) were related to the single-objective inspection stations location problem. The third group of test problems was associated with the multi-objective approach location problem. The preliminary results of applying BCO to the single objective problem of locating inspection facilities in traffic network were considered to be very good. The authors compared these results with the optimal ones for easy and medium examples. In the case of multi-objective optimization various compromise scenarios were generated by choosing different parameters' values. In this way, it was possible to present several feasible alternatives to a potential decision-maker.

3.2.4. Solving the Anti-Covering Location Problem by BCO

The Anti-Covering Location Problem (ACLP) belongs to the class of discrete location problems and could be defined in the following way: For a given set of potential facility location sites, locate a maximally weighted set of facilities in such a way that any two placed facilities are at the distance larger than or equal to some pre-specified value. The total number of facilities to be sited is not given in advance in the case of ACLP. Very important ACLP application encompasses undesirable or obnoxious facilities location (polluting plants, radioactive waste storage sites, explosive storage sites, as well as noise, odor or heat emitters). The spirit of anti-covering restrictions appears as well in the literature examining separation or dispersion of entities, like land management areas, solution selection, franchise distribution, etc.

The authors in [8] demonstrated that the BCO algorithm can be successfully applied to ACLP problem. The BCO implementation was as follows. Within each forward pass, an artificial bee was allowed to choose one node to be a facility location site. Since the number of locations to be located in ACLP was not pre-specified, selecting a single facility was the natural choice. The utility of choosing node i to be a facility site is denoted by V_i. It is inversely proportional to the number of nodes covered by node i. In [8] the utility was calculated as follows:

$$V_i = \frac{C^{max} - C^i}{C^{max} - C^{min}}, \quad i = 1, 2, \ldots, n \tag{10}$$

with

$C^{max} = \left(\max_{i \in N} |N_i \cup i|\right)$ being the largest possible number of nodes covered by any site in current iteration (nodes that are on distance less than or equal to the pre-specified one);

$C^{min} = \left(\min_{i \in N} |N_i \cup i|\right)$ representing the smallest possible number of nodes covered by any site in current iteration;

$C^i = |N_i \cup i|$ - denoting the number of nodes covered by the i-th node in current iteration.

Probability p_i, that a specific bee chooses node i as a facility was equal:

$$p_i = \frac{V_i}{\sum_{k=1}^{n} V_k},\ i = 1, \ldots, n \tag{11}$$

with V_i denoting the utility in the case node i was chosen to be the facility location.

When choosing the next node to be added to the partial solution, artificial bees considered only the available nodes (the nodes not already chosen in the previous partial solutions). The bee also considered only the nodes on distance greater than the specified minimum distance from the current bee's position. In this way, the number of potential choices for each bee was decreasing during the search. During the backward pass, each generated partial solution was evaluated by counting the total number of covered nodes.

The developed BCO algorithm was tested on the given numerical examples. The authors tested 7 examples on the New York network with 30 nodes, 14 examples on the Washington D. C. network with 55 nodes, 9 examples on the London-Ontario network with 150 nodes, and 11 examples on Uganda network containing 152 nodes. All these examples served as the benchmark problems that have been used for testing various approaches on ACLP in relevant literature. The proposed BCO algorithm was capable to find the optimal solution for 39/41 problem instances. At the same time, in the case of 2 problem instances for which the BCO algorithm did not discover the optimal solution, the second best solution was reported.

3.2.5. The BCO approach to the p-Center problem

A new version of the BCO algorithm to deal with the problem of locating p emergency facilities on a network of n vertices (the p-center problem) with symmetric distance matrix was developed in [3]. The p-center problem is a well-known combinatorial optimization problem that belongs to the location theory. It can be defined as follows: Given is the set of n nodes (customers) and distances between any pair of nodes. This structure is usually referred to as network. The goal is to locate p facilities (centers) on a network in such a way to minimize the maximum of the distances from each node to its nearest center. Centers could be located at any of the given n nodes.

As a consequence of unsatisfactory results obtained by applying constructive BCO, the authors of [3] decided to develop a new BCO concept based on improving the complete solution held by each bee. This version of the BCO algorithm was named BCOi. The BCOi implementation for the p-center problem contained the following five steps.

The first step, called preprocessing, performed "off-line", was dedicated to the transformation of the input data in order to reduce the time required for all online computations. Starting from the input distance matrix, three new matrices containing auxiliary data relevant for solution generation were produced.

The second step was the generation of initial complete solutions at the beginning of each BCOi iteration. Initial solution generation was based on the concept of

critical distance, the largest distance among all distances between nodes and their nearest centers. Centers were included in the initial solution one at a time. For each bee, the first center was selected randomly. When considering the remaining nodes as candidates for a new center, current critical distance was identified first (together with the corresponding node and center that define this distance). The circle with the center located in the customer node, defining critical distance and radius equal to the critical distance was constructed (see Fig. 3). Obviously, the only way to improve the current partial solution is to select a node within that circle as the new center. Among these nodes, the authors proposed to select a new center randomly in order to assure the diversity among solutions generated by different bees. The matrices obtained through the preprocessing step were to help performing this selection quickly and easily.

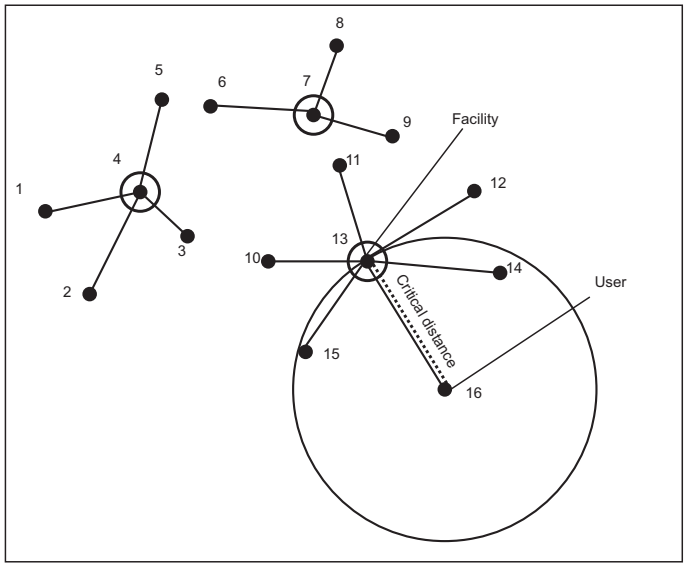

Figure 3: The new center selection process

In the third, the most significant step, the bees modified current solutions. It was repeated NC times (one execution for each forward pass) within the single iteration. It proved to be the key factor for reaching the best possible solution quality. This step was designed in such a way to assure different treatment of the same solutions held upon the recruitment by different bees. Modification consisted of substituting some of the p centers with nodes selected from the remaining $n - p$ non-center nodes. Solution modification was divided into two stages. The first stage was adding q non centers (the solution feasibility was violated) in such a way to reduce the critical distance. In the second stage, q locations were removed from the center list in a greedy manner.

Steps 4 (results comparison mechanism and loyalty decision) and 5 (recruit-

ment) were identical to the corresponding steps from the constructive BCO.

BCOi proposed in [3] was compared with the state of the art techniques used for the p-center problem: multi start interchange, variable neighborhood search, tabu search1 and tabu search2 proposed in [25] and scatter search proposed in [30]. The comparison criteria were solution quality and running time. Comparison was performed on the 40 test examples, originally designed for testing the p-median problem (OR-LibTestProblems [1]). The performed numerical experiments showed that BCOi was able to improve the best known solution for three of the tested examples. It also obtained the best known solutions from the literature for 32 examples (out of 40). For the remaining five examples, BCOi obtained the second best value for the objective function within negligible running time. For almost all these examples (35 out of 40), BCOi running times were smaller than that required by the fastest algorithm from the literature.

3.2.6. The BCO Approach to Optimize Distributed Generation Resources Placement

Using distributed generation resources (DGs) has many advantages such as: distribution loss reduction, line capacity and voltage profile improvement, increasing the reliability, environmentally friendly and etc. Therefore, this problem has been considered intensely in the recent literature. However, despite the aforementioned benefits, the incorrect selection of location, number and/or capacity of DG may lead the rise of network problems. Therefore, to reach optimal operation of distribution network, optimal DG placement and sizing is essential. To meet this goal, the parameters such as losses, line capacity, voltage profile and reliability, should be evaluated in all states before and after the DG installation, and the best solution has to be selected.

The problem of DG placement and sizing for loss reduction and line capacity improvement has been considered and evaluated in [33]. In order to solve this optimization problem, the authors developed constructive BCO approach. The objective function considering two parameters, loss reduction and increasing the free line capacity was defined as follows:

$$F = w_1 F_{ic} + w_2 F_{loss} \tag{12}$$

where w_1 and w_2 represent weights on the importance of each parameter. This function was maximized by the proposed BCO algorithm. The number of bees was set to 100, and as stopping criterion 200 BCO iterations were considered. Also, the number of constructive moves in each forward pass was assumed to be equal to 1. Each constructive move could have two components that are selected randomly: (a) DG location changing and (b) DG power changing. In the objective function, weights were selected as follows: $w_1 = 1$, $w_2 = 4$.

The proposed BCO was tested on sample 33 nodes IEEE network in order to obtain optimal distributed resources allocation. The reported results showed good convergence and stability of BCO: the desired result was obtained in a small number of iterations and each time the program was run, the same output was received.

3.2.7. The BCO Approach to the Capacitated Plant Location Problem

The BCO approach to the well-known capacitated plant location problem was developed in [16]. The considered problem can be formulated in the following way: Given are a set of potential locations for plants with fixed costs and capacities, and a set of customers with demands for goods supplied from these plants. The transportation costs from the plants to each customer are known in advance. The problem is to find the subset of plants that will minimize the total fixed and transportation costs so that demand of all customers can be satisfied without violating the capacity constraints of the plants.

In [16] the authors enriched constructive BCO with a local search routine to enable obtaining good results. Differences from the basic BCO algorithm are in the construction of new solutions. For completing this step in the BCO algorithm, each bee tried to improve the available partial solution a number of times, starting either from the current partial solution ("forager"), or from the new empty solution ("scout"). In the proposed algorithm LBCO (Local search Bee Colony Optimization), these two improvement steps were carried out by different procedures: Forager bees searched for the improvement in the given neighborhood of their current solutions, while scouts choose new solution from the whole feasible region.

At the beginning of each backward pass, the solution quality is determined based on the mathematical formulation of a given problem. Instead by loyalty decision and recruitment, the current set of solutions was updated in such a way that α best and β worst solutions are replaced by the new ones. New parameters α and β determining the relevant subsets of good and bad solutions were introduced. Their values were updated in each backward step based on the quality of the current set of solutions compared with the average value of the qualities of all solutions retrieved so far.

The authors compared LBCO results with those obtained by Ant Colony Optimization on some well known test examples. They used two types of test instances. The first one, from OR-Library [1], was found to be very easy for both LBCO and ACO. For all such problems, the optimal solutions have been found with negligible CPU times. On test examples from the electronic library of the Sobolev Institute of Mathematics of the Siberian Branch of the Russian Academy of Science, the LBCO algorithm has shown better results in comparison with ACO from the CPU time point of view.

3.3. Application of BCO to Scheduling Problems

3.3.1. Scheduling Independent Tasks to Identical Machines by BCO and Parallel BCO

The problem of static scheduling independent tasks on identical machines can be described as follows. Let $T = \{1, 2, \ldots, n\}$ be a given set of independent tasks, and $M = \{1, 2, \ldots, m\}$ a set of identical machines. The processing time of task i ($i = 1, 2, \ldots, n$) is denoted by l_i. All tasks are mutually independent, and each task can be scheduled to any machine. All given tasks should be executed. A task should be scheduled to exactly one machine, and machines can execute one task

at a time. The goal is to find the scheduling of tasks to machines in such a way as to minimize the completion time of all tasks (the so called makespan).

The problem was treated in [6, 7] by the BCO heuristic algorithm. In each of its iterations, BCO performed constructive steps composed of forward and backward passes and within them generated B solutions (schedules), one schedule for each bee. Within each forward pass, every artificial bee was allowed to generate NC task-machine pairs. The probability that task i would be chosen by any bee was denoted by p_i and was calculated as follows:

$$p_i = \frac{l_i}{\sum_{k=1}^{K} l_k},\ i = 1, \ldots, n \tag{13}$$

where:

l_i - is the processing time of the i-th task;

K - represents the number of "free" tasks (not previously chosen).

Obviously, tasks with longer processing times had higher chances to be chosen. The probability p_j of choosing machine j by any bee was equal to:

$$p_j = \frac{V_j}{\sum_{k=1}^{m} V_k},\ j = 1, \ldots, m \tag{14}$$

where:

$$V_j = \frac{\max F - F_j}{\max F - \min F},\ j = 1, \ldots, m \tag{15}$$

with:

F_j being running time of a machine j based on tasks already scheduled to it;

$\max F$, $\min F$ representing maximum and minimum over all machines running times.

Obviously, V_j represented the normalized value for the running time of corresponding machine and was used in the definition of probability for its selection. The backward pass started with the evaluation of all partial solutions generated during the preceding forward pass. The latest time point of finishing the last task at any machine characterized each generated partial solution (Fig. 4). After solutions were evaluated (and normalized), the loyalty decision and recruiting process were performed in the usual way.

The BCO implementation was tested on examples with known optimal solutions. Optimal solutions were obtained by using ILOG AMPL and CPLEX 11.2 optimization software. Both programs, CPLEX and BCO were running on the same computer, and the authors were able to compare execution times and solution quality for these programs.

The proposed BCO algorithm was able to obtain the optimal value of objective function in all test problems. The CPU times (around 1 sec.) and the number of iterations (usually, less than 10) required to find the best solutions by the BCO were very small. The BCO algorithm parameters were set to the following values: $B = 5$; $NC = 10$; and $I = 100$ iterations selected as the stopping criterion.

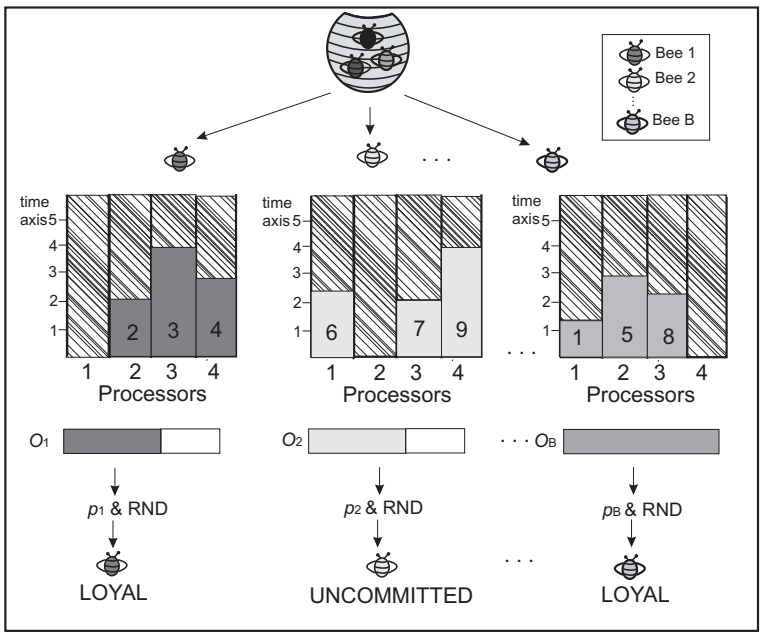

Figure 4: Comparison of partial solutions after the third forward pass, $NC = 1$

In order to improve results obtained by [6], the authors embedded global knowledge in the search process in [7]. They allowed the relevant information to be available through different iterations and to be explored by bees during the solution generation process. The current global best objective function value (makespan) was 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. More precisely, a bee would not allocate task i to processor j in the situation where this would exceed the current global best makespan. More precisely, the probability p_j given by (14) would be set to zero for each processor j satisfying this condition. In the rare cases when all probabilities should be set to zero, a processor was selected randomly in order to complete the current forward pass. Regarding the backward pass, the same information was used to prevent bees staying loyal to partial solutions inferior to the current global best objective function. Consequently, the loyalty probabilities for those bees were set to zero and they automatically became uncommitted. If all bees become uncommitted in this way, the iteration was interrupted and a new one started without completing the current partial solutions. The modified version of BCO outperformed the original one regarding both scheduling results and running time. In addition, applied to Bin Packing Problem (BPP), another similar combinatorial optimization problem, the modified BCO outperformed two state-of-the-art techniques with respect to solution

quality as well as execution time for smaller examples. For larger examples BCO performed better than Ant Colony Optimization (ACO) from [17]], while Hybrid Grouping Genetic Algorithm (HGGA) proposed in [11] was capable of generating solutions of higher quality. However, BCO was not originally developed for BPP while HGGA represents a hybrid method that includes local search procedure.

The additional improvement of scheduling results was obtained by parallelization of the BCO algorithm [2, 4]. Five coarse-grained parallel implementations for BCO under the Message Passing Interface (MPI) were proposed and tested on the given scheduling problem. The first strategy assumed independent execution of various BCO algorithms. Sequential versions of BCO were executed on different processors independently and the best solution was collected at the end. For a modest number of engaged processors (≤ 12), the obtained speedup was almost linear, with the same quality of the final solution or degraded for a small amount (below 3% with respect to the sequential result).

Two synchronous cooperative variants were also implemented on a completely connected homogeneous multiprocessor system, in which processors communicated by exchanging messages. The variant involving less frequent knowledge exchange resulted in better performance: the quality of the solution and/or minimum running time was improved for a modest number of engaged processors. In addition, two variants of asynchronous parallel BCO were implemented. The first of them included a global memory concept, and it was implemented on master-slave multiprocessor architecture. The second, a non-centralized asynchronous execution was realized on a unidirectional processor ring. On two hard test examples, it was shown that, while both the synchronous and asynchronous cooperation concepts performed well on a modest number of processors, the asynchronous concept outperformed the synchronous one as the number of engaged processors increased.

3.3.2. Solving the Ride-Matching Problem by BCO

Urban road networks in many countries are severely congested, resulting in the increased travel times, the increased number of stops, unexpected delays, greater travel costs, inconvenience to drivers and passengers, increased air pollution and noise level, and/or increased number of traffic accidents. Transportation professionals have developed different Travel Demand Management (TDM) strategies to solve these problems. Ride-sharing is one of the widely used TDM techniques. Within this concept, two or more persons share a vehicle when travelling from their origins to the destinations. The operator of the system must possess the following information regarding trips planned for the next week: (a) Vehicle capacity (2, 3, or 4 persons); (b) Days in the week when person is ready to participate in ride-sharing; (c) Trip origin for every day in a week; (d) Trip destination for every day in a week; (e) Desired departure and/or arrival time for every day in a week.

The ride-matching problem considered in [36, 37] could be described in the following way: Make routing and scheduling of the vehicles and passengers for the whole week in such a way to minimize the total distance travelled by all

participants. The authors developed BCO based model for the ride-matching problem and started their choice model from the assumption that the quantities perceived by bees are "fuzzy" [46]. The implemented BCO used approximate reasoning and rules of fuzzy logic in the communication between bees and the corresponding decision making.

When adding the solution component to the current partial solution during the forward pass, each bee perceived a specific solution component as 'less attractive', 'attractive', or 'very attractive'. Artificial bee could perceive some other specific attributes as 'short', 'medium' or 'long'; 'cheap', 'medium', or 'expensive'; etc. The authors developed the approximate reasoning algorithm for calculating the solution component attractiveness. In order to describe bee's partial solutions comparison mechanism, the authors introduced the concept of partial solution badness. The partial solution badness was calculated in the following way:

$$L_k = \frac{L^{(k)} - L_{min}}{L_{max} - L_{min}} \tag{16}$$

where:

L_k represented the badness of the partial solution discovered by the k-th bee;

$L^{(k)}$ was the objective function value of the partial solution discovered by the k-th bee;

L_{min} and L_{max} denoted the objective function value of the best and the worst partial solution discovered from the beginning of the search process, respectively.

The approximate reasoning algorithm to determine bee's loyalty to its partial solution contained the rules of the following type:

If the discovered partial solution is BAD
Then loyalty is LOW

The bees used approximate reasoning and compared their discovered partial solutions with the best and the worst discovered partial solution from the beginning of the search process. In such a way, 'historical facts' discovered by all members of the bee colony significantly influenced the future search directions.

Based on the quality of its solution, each bee decided with certain probability weather to stay loyal or to become an uncommitted follower. Each partial solution (partial path) being advertised in the dance area had two main attributes: (a) the objective function value; and (b) the number of bees that were advertising that partial solution (partial path). The number of bees advertising the partial solution was a good indicator of a bees' collective knowledge. It showed how a bee colony perceives specific partial solutions. The authors used the approximate reasoning algorithm to determine the advertised partial solution attractiveness. It contained rules of the following type:

If the length of the advertised path is SHORT
and the number of bees advertising the path is SMALL
Then the advertised partial solution attractiveness is MEDIUM

The approximate reasoning algorithm was used to calculate the number of shifting bees with the rules of the following type:

If bees' loyalty to path p_i is LOW
and the attractiveness of path p_j is HIGH
Then the number of shifting bees from path p_i to path p_j is HIGH

In this way, before the beginning of the new forward pass, the number of bees considering a specific path was determined. Using collective knowledge and sharing information, bees concentrated on more promising search paths

Proposed model was tested in the case of real-life ride-sharing demands from Trani, a small city in the south-east of Italy, to Bari, the regional capital of Puglia. The authors collected data regarding 97 travelers demanding ride sharing, and assumed, for the sake of simplicity, that capacity is four passengers for all their cars. The results obtained by BCO showed that proposed algorithm had converged to optimal solution (optimal schedule of passengers into cars) for less than 120 iterations.

3.3.3. BCO Approach to the Backup Allocation Problem

In the paper [27] backup allocation problem (known as "data backup" process) was considered. This problem could be classified as a scheduling problem. The computer networks providing on-line services to the consumers can increase the quality of their services establishing backup data stored on network servers. In such way, a "cold reserve" of the data is created enabling the recovery in the case of hardware and/or software failure, thus providing continual service to the users. The input data for this problem are the total number of services which are subject to backup process, and their parameters related to the amount of data, their complexity, and the estimated transfer rate (backup intensity). The objective function is related to the minimization of the duration of the entire backup process.

The main characteristics of the BCO application on backup allocation problem proposed in [27] were:

- In a given server network, all nodes represented services.
- Hive was an artificial node which did not influence the search process.
- The authors used LOGIT model to calculate probability that a particular node would be chosen for server allocation.
- The solution quality was defined as the total time spent on the backup data.

The authors of [27] used a logical network of servers with 14 different services ready for backup process on these servers. It was assumed that the space needed for the backup is available at the time required for the process. The main goal was to finish backup process in the shortest possible time period when there was no significant computing activity on the computer network. The authors empirically concluded that this is the time between 3 and 5:30 am.

The application of BCO required 2.56h for performing a complete backup which is a bit longer than the desired period (2.5h). In practice, however, the specified interval was acceptable because it did not stand out dramatically from the defaults, and, on the other hand, a small number of active system users in the interval were not affected dramatically with the backup process.

3.3.4. Scheduling for Computational Grid by BCO

The efficient scheduling of the independent and sequential tasks on distributed and heterogeneous computing resources within grid computing environments is an NP-complete problem. Grid computing is a large scale distributed environment designed for solving the computational and data-intensive problems in science and industry. Actually, the grid is an infrastructure which supplies a mechanism to run the applications over computational resources which are heterogeneous and geographically distributed. Grid managers apply task scheduling algorithms to dispatch the submitted tasks among the grid resources appropriately. Generally, the grid tasks are submitted to the grid managers by grid users, and then the manager schedules the tasks within the available resources (Fig. 5).

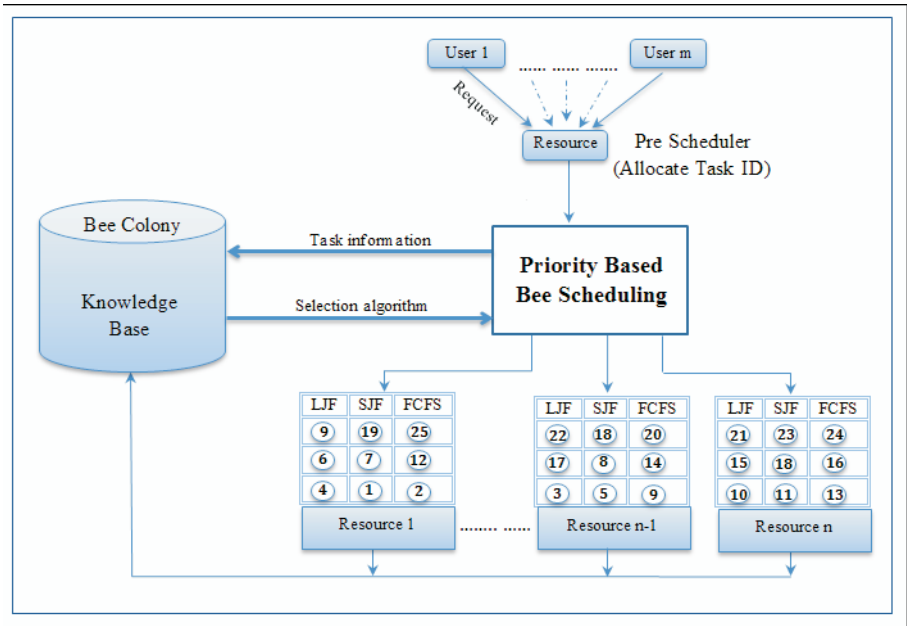

Figure 5: The general scheme of the proposed algorithm

The authors of [26] proposed a new task scheduling algorithm based on the BCO approach. The scheduling of the independent and sequential tasks was done in similar way as the authors did in [6]. The main differences between these two approaches are:

- Task priorities and deadlines were specified as the input data.
- A unique ID was assigned by pre-scheduler to each task before it was sent to a resource named Priority Based Bee Scheduling.
- The Knowledge Base using the bee colony algorithm was created. It in-

cluded a table with the following data for each task: ID, arrival time, priority, deadline, data size, start time, execution time, finish time, delay time, execution algorithm and the destination resource allocated to the task.

The role of each bee during the forward pass was to randomly select a scheduling algorithm for each task and to determine its completion time based on the produced output. The possible scheduling algorithms were First Come First Served (FCFS), Shortest Job First (SJF) and Longest Job First (LJF). During the backward passes, the most appropriate schedule for each task was determined by comparing all schedules generated by the bees. The main steps of the proposed algorithm are illustrated below [26].

- **While** there is any unscheduled task do
  1. Select task T based on maximum priority and minimum deadline and then update tasks table,
  2. **For** each of the resources calculate the computation and data transmission times and then compute the task completion time,
  3. Select the minimum completion time,
  4. **Until** the number of the tasks existing in the training set is less than the pre-specified number **do**
     a. Randomly select the scheduling algorithm among the given ones,
     b. Send the information of task T to the Knowledge Base,
     c. **Go to** step 7.
  5. **For** each task (or bee) do the forward pass (in the initialization phase, randomly select one of the given scheduling algorithms, and assign it to the task),
  6. **For** each task do the backward pass:
     a. Send the information of task T to the Knowledge Base (This step is equal to the returning of all of bees to the hive),
     b. Sort the tasks' table by the value of tasks' objective functions,
     c. Choose a suitable algorithm for each of the tasks (based on the objective function of that task) which results in the lowest finishing time,
     d. Each task decides with probability to continue its own algorithm.
  7. Attach the selected algorithm to task T,
  8. Send task T to the selected queue on the resource R,
  9. Execute task T on the resource R based on the nearest deadline,
  10. Update Knowledge Base after finishing the execution of the task.
- **End While**.

The authors measured the performance of the proposed algorithm by comparing it with the selected scheduling algorithms (FCFS, SJF, LJF) working individually. Various case studies have been considered. The obtained simulation results showed the dominance of the presented algorithm based on BCO over the selected scheduling algorithms. Applying the proposed algorithm to the grid computing environments, the maximum delay and finish times of the tasks were reduced. Namely, the total makespan of the environment was minimized and the

deadline and priority requirements of the tasks were satisfied.

3.3.5. The BCO Approach to Job Shop Scheduling Problem

The deterministic Job Shop Scheduling Problem (JSP) consists of a finite set J of n jobs to be processed on a finite set M of m machines. Each job J_i must be processed on every machine and consists of a chain of m_i operations $O_{i1}, O_{i2}, \ldots, O_{im}$, which have to be scheduled in a pre-determined given order. O_{ij} is the j-th operation of job J_i which has to be processed on a machine M_x for a processing time period of τ_{ij} without interruption and preemption. Each machine can process only one job and each job can be processed by only one machine at a time. The longest duration in which all operations of all jobs are completed is referred to as the makespan C_{max}. In [31] the authors noticed that when the basic constructive BCO [35] was applied to job shop scheduling problem, the improvement of the current best solution was too slow and BCO could easily be trapped in a local minimum.

In order to improve its performance, three modifications of BCO were proposed yielding a new method named MBCO. The proposed modifications included global evolution for some bees, dynamic changes of parameters, and special treatment for the current best solution. Global evolving was implemented by allowing some good (partial) solutions from current iteration to be exploited in the next iteration. MBCO used dynamic parameters, i.e., the number of constructive moves and the probability to keep a solution, changed their values from iteration to iteration. In addition, a special improvement move, using Tabu Search, was performed to the best solution produced by a bee.

a) Global evolution

A fixed number of bees started a new iteration from the initial parts of some good complete solutions from the previous iteration. In the next iteration, these bees tried to improve final solutions that could be generated starting from the given initial part. The bees were expected to find the complete solution that was as good as or even better than the previous one.

b) Dynamic Parameters

Contrary to the original BCO from [35], MBCO used different NC in each iteration, which was randomly generated from the predefined interval $[NC_{min}, NC_{max}]$. This allowed the bee colony to find solutions using different strategy in each iteration. The idea was to increase the solution diversity and to reach a better solution more quickly. The authors introduced some additional dynamic parameters for calculating the probabilities in loyalty decision process. The goal of this modification was guiding the bees to focus on the area closer to the optimum solution.

c) Special Treatment for the Best Solution

The third modification was to combine the foraging behavior in BCO with other bee's intelligent behavior. In nature, a queen is the biggest bee in the colony developed from a selected larva that gets more and better food than the others. So, the queen candidate grows to be the best bee in the colony. That behavior was adopted to improve the performance of BCO. In MBCO, an artificial bee producing the best solution was allowed to improve it by a local search. The

improved solution was expected to be the new best solution and its initial par could be used in the next iteration. In [31] Tabu search (TS) was used as the local search technique.

Ten instances of JSP were used to compare the performances of BCO and MBCO. MBCO resulted in higher accuracy than the original BCO.

3.3.6. BCO for Minimum Cost Berth Allocation Problem

Within the Berth Allocation Problem (BAP) given are the berth layout of a container terminal and a set of vessels which are to be served within the planning time horizon. Vessels are represented by a set of data including the expected time of arrival, the size, anticipated handling time, a preferred berth in the port, and penalties, along with other information. BAP can be defined as follows: for each vessel in the set, the berth index and the time interval are allocated in such a way that a given objective function is minimized. If the objective is the minimization of berthing cost as well as the costs of earliness and delay of each vessel, we are dealing with the Minimum Cost BAP. This problem has been proven to be a NP-hard problem. If it is assumed that both coordinates are discrete (space is modeled by the berth indices whereas the time horizon is divided into segments in such a manner that berthing time of each vessel can be represented by an integer), a solution to BAP can be described in a space-time-diagram as it is illustrated in Fig. 6. Each vessel is represented by a rectangle, whose height suits the length of a vessel (expressed by the number of berths), and the width matched the required handling time. The berthing position and berthing time of a vessel is denoted in the lower-left vertex of a rectangle and is referred to as the *reference point* of a vessel (marked by the index of vessel in Fig. 6). A berth plan is *feasible* provided that the rectangles fit the time-space diagram and do not overlap (see Fig. 6).

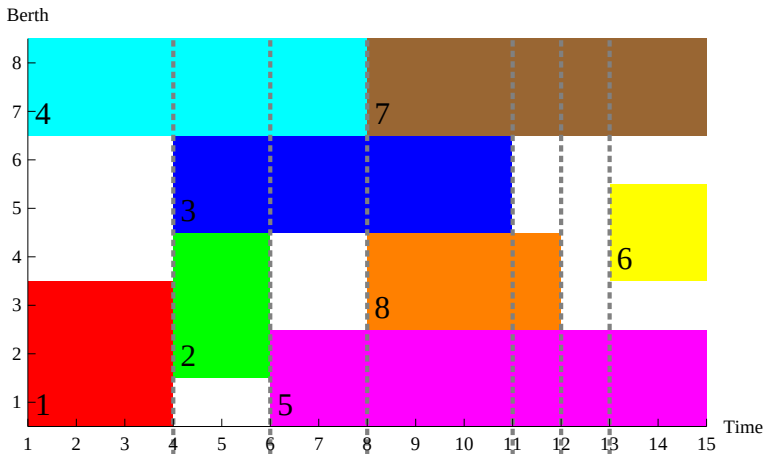

Figure 6: Output of BCO for BAP solution

The BCO approach to minimum cost BAP was proposed in [15]. A combination of constructive and improvement BCO was developed in the following way. During the off-line initialization phase, for all vessels and for all possible positions of the vessel in 2-dimensional plane, the list (named ξ list) of 3-tuples elements (berth, time, penalty cost) was created. The list associated to a vessel was sorted in the increasing order according to the penalty cost value. *NC* parameter value was set to $l/2$ (two vessels were added to the solution in each forward pass), and *B* took value 10. Each bee was assigned its own ξ list containing an empty solution at the beginning of each iteration. In order to create its partial solution, a bee had to make two decisions: which vessel to choose and where in two-dimensional plane to place the selected vessel. These decisions were taken using the randomly generated number and roulette wheel. Once the vessel was added to the partial solution, the corresponding bee reduced its ξ list by removing conflicting positions for the unused vessels.

Three types of improvement steps were proposed in [15]. Each bee holding a complete solution tried to improve it at the end of each iteration by rearranging vessels to non-conflicting positions with smaller costs. This represented the first type of the improvement. The other two improvements were performed only to the current global best solution due to their computational complexities. Both of them required the rearranging ξ list. In the second improvement, a search for feasible positions with smaller costs for each vessel was performed, while the third improvement involved also the resolving conflicting situations. This improvement was performed after *NC* iterations.

The proposed BCO was compared with CPLEX commercial software and MIP-based meta-heuristic methods. The real-life test instances characterized by 21-28 vessels, 12 berths and 54 time units within the time horizon were used for the comparison. With the application of the BCO meta-heuristic, better results are achieved regarding quality as well as of CPU time spent to find the final solutions. In all test examples, BCO was able to generate the optimal solutions within the CPU time 100 times shorter than MIP-based meta-heuristic approach, and 300 times shorter than the time needed by CPLEX.

3.4. Application of BCO to Medical and Chemical problems

3.4.1. Dose Planning in Well Differentiated Thyroid Cancer Treatment

Thyroid cancers are the most common endocrine carcinomas. Various clinical parameters (the patient's diagnosis, the patient's age, the tumor size, the existence of metastases in the lymph nodes and the existence of distant metastases) influence a physician's decision in dose planning. The weights (importance) of these parameters were determined in [38] by the BCO meta-heuristic. In addition, the Case-Based Reasoning (CBR) was used to describe a physician's expertise, intuition and experience when treating patients with well differentiated thyroid cancer.

In [38] an improvement version of BCO was used as a tool. At the beginning of the process, complete solutions were generated randomly, i.e., all of the weights were assigned to each solution in a random manner. After finishing the recruiting

process, the loyal bees had the same values for weights as at the beginning of the previous forward pass. The recruiter's weight values were copied to a solution of an uncommitted follower. The recruiter's solution became the follower's, and after that, bees applied different modification steps to the same solution.

At the beginning of the new forward pass, the parameters' weights were changed by the solution modification process. In [38] the authors randomly chose a specific weight and reduce it by 0.1. This procedure was repeated by alternating increasing and reducing values of randomly chosen weights by 0.1. In this way, a diversification process was provided, taking into account that the sum of the weights must be equal to 1 and each weight must be a positive number.

The proposed CBR-BCO model was used to suggest the I-131 iodine dose in radioactive iodine therapy. It was tested on real data from patients treated in the Department of Nuclear Medicine, Clinical Center Kragujevac, Serbia. By comparing the results obtained by the developed CBR-BCO model to the physician's decision, the authors concluded that the CBR-BCO was highly reflecting the reality. The presented CBR-BCO model made the same decisions as the physician in 87.5% tested cases (35 out of a total of 40 examples).

3.4.2. Optimizing chemical processes by BCO

The efficiency of BCO in optimizing a chemical process was demonstrated in [32]. Catalytic reforming of naphtha is an important process in the refining industry for octane improvement or as a source of aromatics and hydrogen. The aim of authors was to introduce and apply the BCO algorithm for optimizing operating conditions of Continuous Catalytic Regenerative (CCR) process reactors. Optimization results were compared with those obtained by Genetic Algorithm.

The approach proposed in [32] represented a modification of the basic BCO. The authors distinguished two types of bees, the scout bees and the forager bees. The scouts were performing random search in order to generate the so called "average solution", the one whose profitability would be considered as a threshold for loyalty decision of other bees. If the profitability of the solution found by the forager in the forward pass was greater than the expectations of the entire colony, the bee stayed loyal and advertised its solution. If the solution profitability is smaller than the average, the bee would become a follower.

For the algorithm used in [32], a new parameter was introduced: *FN*-Number of food sources (sets of neighboring solutions). The BCO algorithm started with randomly producing food source sites that corresponded to the solutions in the search space, according to the boundaries of the variables. For producing initial food source sites (initial solutions), the following random function was used:

$$x_{i,j} = x_j^{lower\ bound} + rand(0,1)\left(x_j^{upper\ bound} - x_j^{lower\ bound}\right) \tag{17}$$

where $i = 1,\ldots,B$ and $j = 1,\ldots,D$ and roulette wheel method was used for producing the initial solutions of bees. D denoted the total number of variables in the considered problem.

Each bee was associated with only one food source site and produced a modification of the solution in its memory, depending on its neighboring food sources and local information.

The authors used three sets of industrial data for determining kinetic model parameters and model validations. The BCO algorithm was used for optimizing the CCR process. The results were compared with the values obtained by GA and industrial data. It was shown that the BCO algorithm outperformed GA significantly. BCO was found to be considerably faster than GA. The authors concluded that the BCO algorithm can be successfully applied for optimizing chemical engineering processes.

3.5. Network Problems

3.5.1. Network Design

Properly designed public transit network can significantly increase public transport mode share. This transportation planning problem belongs to the class of difficult combinatorial optimization problems, whose optimal solutions are hard to be discovered. The bus network shape as well as bus frequencies highly depend on several factors: passenger demands, number of available buses (fleet size), their type, and/or available budget.

The model for the bus network design problem was developed in [29]. The authors approached this problem by designing the BCO based method. Their goals were to maximize the number of served passengers, to minimize the total in-vehicle time of all served passengers, and to minimize the total number of transfers in the network. The specific characteristics of the proposed BCO algorithm tailored for the network design problem were:

- The BCO algorithm based on the improvement concept was used. In other words, the bees investigated solution space in the neighborhood of the current solution, and tried to improve their solutions. The modification of a solution was performed through $NP = 5$ forward passes within the single iteration.
- At the beginning of a network design, all artificial bees were supposed to be in the hive. The hive represented an artificial location, not connected to either of the bus lines.
- The initial solution was generated by the simple greedy algorithm.
- The major difference comparing to the previous BCO applications was the existence of heterogeneous bees. Two sets of artificial bees were used to solve the transit network design problem. The main difference between two types of bees was in the solution modification process. When making decisions about the loyalty, as well as during the recruiting process, both bees of type 1 and bees of type 2 behave in the same way.
- The equation for calculating the loyalty probability was simplified. The difference between O_{max} (the normalized value of the best objective function

value among all bees' solutions) and O_b values was not divided by u (the counter (ordinary number) of the forward pass). In such a way, the authors didn't take into account the influence of efforts already made within the current iteration on bees' decisions.

- A new parameter was introduced in order to distinguish: NP - the number of forward and backward passes in a single iteration, and NC - the number of changes in one forward pass. For the network design problem NC was set to 2. Up to now, in all improvement versions of BCO, the bees performed a single solution modification.

The numerical experiments were performed on both the known benchmark examples and the problems generated by the authors. The BCO approach was compared with other state-of-the-art approaches from the literature and was superior in 3 out of 4 cases.

3.6. Continuous and Mixed-Optimization Problems

3.6.1. Numerical functions minimization by BCO

Empirical study of the BCO algorithm was performed in [28]. The authors applied BCO to optimize the set of well known numerical test functions. In continuous optimization, the main goal is to find the global minimum of a function, i.e., to discover the smallest value of the function in the given range. The obtained results were compared with the results achieved by the Artificial Bee Colony (ABC), Genetic Algorithm (GA), Differential Evolution (DE), and Particle Swarm Optimization (PSO)

Due to the empirical testing of the proposed algorithm, the authors decided to use the improving version of BCO. At the beginning of BCO execution, the initial solutions were generated in a random manner, and then, by perturbing solutions, bees improved them through iterations. The main difference between the previous approaches and the BCO proposed in [28] is the propagation of the best known solution through modification process in the later BCO. The best among randomly generated initial solutions was set to be the current best (incumbent) solution. At the beginning of each iteration, the authors took into account the incumbent, as well as B solutions created within the previous iteration ($B + 1$ solutions in total). By using the roulette wheel selection, each bee chose one among $B + 1$ considered solutions to be the initial solution for the next iteration.

To properly deal with the optimization of continuous test functions on n variables, the authors introduced three new parameters: LB - left boundary (the minimum value of the variable), RB - right boundary (the maximum value of the variable), and d - range (initially set to the difference between LB and RB). The value of d was decreased in each iteration, up to the lower bound equal to 0.001. The values of these n variables were changing in the following way:

- Randomly choose the number of variables to be modified.
- Randomly choose the variables to be changed.

- Randomly decide whether to decrease or to increase the value of each chosen variable.

In [28] the BCO approach was applied to the minimization of 51 benchmark functions from the literature. The numerical experiments showed the superiority of the BCO algorithm over other methods used in the comparison. Comparing with GA, BCO performed better in 34 (out of 51) cases, in 13 examples there was no statistically significant difference between BCO and GA, while GA outperformed BCO only for 3 examples. BCO and PSO performed the same in 27 examples, and on the remaining instances BCO won 22:1. BCO outperformed DE 14:1, with similar performance noticed for the remaining 35 cases. Finally, when comparing BCO and ABC, there was no significant difference between them for 34 functions. BCO performed better in 12, while ABC won only for the remaining 4 cases.

3.6.2. BCO for the satisfiability problem in probabilistic logic

Satisfiability problem (SAT) is the first optimization problem for which *NP* completeness has been proven. It is a well studied optimization problem due to its significance in logic, computer science and many other fields. A less studied variant of this problem, the satisfiability in probabilistic logic (PSAT) is important for the formalization of reasoning with uncertainty. PSAT can be easily reduced to a linear programming problem, however, solving it by any standard linear solver is inapplicable in practice due to the complexity of the problem. Therefore, meta-heuristics have been applied in the relevant literature. A variant of PSAT, the satisfiability problem in approximate conditional probabilities (CPSAT-ε) was considered in [34]. The main differences between PSAT and CPSAT-ε are: (a) CPSAT-ε involves conditional probability operator on the contrary to PSAT, and (b) probabilities of formulas in CPSAT-ε may take infinitesimal values, contrary to the PSAT where probabilities are real-values. The CPSAT-ε reduces to a linear program over probabilities calculated for each of the atoms. If the formula can be satisfied, i.e., the solution to CPSAT-ε exists, the number of atoms with nonzero probability values is equal to $L + 1$, where L is a number of inequalities in the system. The solution is therefore an array containing $L + 1$ atoms

$$x = [a_1, a_2, \ldots, a_{L+1}],$$

with assigned probabilities

$$p = [p_1, p_2, \ldots, p_{L+1}].$$

The probabilities of atoms not in x are assumed to have zero values.

In [34] the BCO approach based on the improvement concept was proposed to find a model (valuations of binary variables and corresponding conditional probabilities) for the tested formulae. This was the first application of BCO to a class of problems involving search for a solution (contrary to the situations where the problems already have some feasible solutions that one wants to improve). An additional reason for selection of BCO to deal with CPSAT-ε was its ability

to allow both the improvement and the degradation in solution quality. Local search based methods are easily trapped in the neighborhoods of the current best solution that may not lead to the global best, i.e., to the solution that satisfies the considered formula. On the other hand, evolutionary methods involve more randomness that may diversify the search and require more time to get to the desired final solution. The implementation of BCO proposed in [34] proved to be the right choice.

The BCO implementation proposed in [34] consisted of the following four phases. The first phase was the generation of the initial solutions at the beginning of each iteration. The 5(L+1) atoms were chosen randomly. To each atom equal probabilities, i.e., 1/(L+1), were assigned and the grades were calculated. From these atoms, the (L+1) with the best grades were selected to form the initial solution. The second phase was devoted to the solution modification. During the solution modification process, the probabilities of selected atoms were changed in order to obtain the combination that represents the solution of a given system. Two heuristics known from the literature (worst unsatisfied projection and greedy giveaway) were used to reduce the infeasibility. In the phase 3 solution comparisons mechanism was performed, while recruitment was a subject of phase 4.

The experimental results obtained by testing CPSAT-ε were compared with those obtained by using the exact Fourier-Motzkin elimination procedure and thus demonstrated the superiority of the BCO method.

4. CONCLUSION

The Bee Colony Optimization (BCO) algorithm, a meta-heuristic method inspired by the foraging behavior of honeybees, belongs to the class of Swarm Intelligence techniques. It represents a general algorithmic framework applicable to various optimization problems in combinatorial/continuous optimization, and engineering. The BCO method is based on the concept of *cooperation*, which increases the efficiency of artificial bees and sometimes even allows achieving goals that could not be reached only by individual actions. Through the information exchange and recruiting process, BCO has the capability to intensify the search in the promising regions of the solution space. BCO has become very popular algorithm due to its simplicity: it is easy to understand and has a small number of parameters (number of bees and number of transformations during a single iteration).

This paper reviews the existing applications of the BCO algorithm to several combinatorial and continuous optimization problems in order to promote this simple yet effective optimization method. The survey is certainly not exhaustive, and we hope that expanded application reports are to come soon. Moreover, the suitability for parallelization of BCO opens not only a new research direction but also some new potential applications. Based on the achieved results and gained experience, new models founded on BCO principles (autonomy, distributed functioning, self-organizing) are likely to significantly contribute to solving complex engineering, management, and control problems. 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.

References

[1] J. E. Beasly. A note on solving large *p*-median problems. *European Journal of Operational Research*, 21 (1985) 270–273.

[2] T. Davidović, T. Jakšić, D. Ramljak, M. Šelmić, and D. Teodorović. Mpi parallelization strategies for bee colony optimization. *Optimization, Special Issue entitled "Advances in Discrete Optimization" dedicated to BALCOR 2011*, 62(8) (2013) 1113–1142. DOI:10.1080/02331934.2012.749258.

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

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

[5] T. Davidović, D. Teodorović, and M. Šelmić. Bee colony optimization Part I: The algorithm overview. *Yugoslav Journal of Operational Research*, 25(1) (2015) 33–56.

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

[7] T. Davidović, M. Šelmić, D. Teodorović, and D. Ramljak. Bee colony optimization for scheduling independent tasks to identical processors. *J. Heur.*, 18(4) (2012) 549–569. DOI:10.1007/s10732-012-9197-3.

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

[9] L. Duckstein. Multiobjective optimization in structural design:the model choice problem. In *New Directions in Optimum Structural Design*, John Wiley and Sons, New York, (1984) 459–448.

[10] P. Edara, J. Guo, B. L. Smith, and C. McGhee. Optimal placement of point detectors on Virginia's freeways: case studies of northern Virginia and Richmond. Technical report, Final Contract Report VTRC 08-CR3, Virginia Transportation Research Council, Richmond, VA, 2008.

[11] E. Falkenauer. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristic*, (2) (1996) 5–30.

[12] S. L. Hakimi. Optimal locations of switching centers and the absolute centers and medians of a graph. *Operations Research*, 12 (1964) 450–459.

[13] S. L. Hakimi. Optimum distribution of switching centers in a communication network and some related graph theoretic problems. *Operations Research*, 13(3) (1965) 462–475.

[14] O. Kariv and S. L. Hakimi. An algorithmic approach to network location problems. Part II: The *p*-median. *SIAM Journal of Applied Mathematics*, 37 (1979) 539–560.

[15] N. Kovač. Bee colony optimization algorithm for the minimum cost berth allocation problem. In *XI Balkan Conference on Operational Research*, (BALCOR, 2013), Beograd–Zlatibor, (2013) 245–254.

[16] T. V. Levanova and E. A. Tkachuk. Development of a bee colony optimization algorithm for the capacitated plant location problem. In *II International conference, Optimization and applications (OPTIMA-2011)*, Petrovac, Montenegro (2011) 153–156.

[17] J. Levine and F. Ducatelle. Ant colony optimization and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society*, 55 (2004) 705–716.

[18] P. Lučić and D. Teodorović. Bee system: modeling combinatorial optimization transportation engineering problems by swarm intelligence. In *Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis*, Sao Miguel, Azores Islands, (2001) 441–445.

[19] P. Lučić and D. Teodorović. Transportation modeling: an artificial life approach. In *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence*, Washington, DC, (2002) 216–223.

[20] P. Lučić and D. Teodorović. Computing with bees: attacking complex transportation engineering problems. *International Journal on Artificial Intelligence Tools*, 12(3) (2003) 375–394.

[21] P. Lučić and D. Teodorović. 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.

[22] G. Marković, D. Teodorović, and V. Aćimović-Raspopović. Routing and wavelength assignment in all-optical networks based on the bee colony optimization. *AI Commun.*, 20(4) (2007) 273–285.

[23] D. McFadden. Conditional logit analysis of qualitative choice behavior. In P. Zaremmbka, editor, *Frontier of Econometrics*. Academic Press, New York, (1973) 105–142.

[24] N. Mladenović, J. Brimberg, P. Hansen, and J. A. Moreno–Pérez. The *p*-median problem: A survey of metaheuristic approaches. *European Journal of Operational Research*, 179(3) (2007) 927–939.

[25] N. Mladenović, M. Labbe, and P. Hansen. Solving the *p*-center problem with tabu search and variable neighborhood search. *Networks*, 42(1) (2003) 48–64.

[26] Z. Mousavinasab, R. Entezari-Maleki, and A. Movaghar. A bee colony task scheduling algorithm in computational grids. In *Digital Information Processing and Communications, Proc. Int. Conf., ICDIPC 2011, Part I*, Springer Berlin Heidelberg, Ostrava, Czech Republic, (2011) 200–210.

[27] R. Nedeljković, S. Mitrović, and D. Drenovac. Bee colony optimization meta-heuristic for backup allocation problem. In *Proc. PosTel XXVII*, (in Serbian), Beograd, Serbia, (2009) 115–122.

[28] M. Nikolić and D. Teodorović. Empirical study of the bee colony optimization (BCO) algorithm. *Expert Systems with Applications*, 40(11) (2013) 4609–4620.

[29] M. Nikolić and D. Teodorović. Transit network design by bee colony optimization. *Expert Systems with Applications*, 40(15) (2013) 5945–5955.

[30] J. A. Pacheco and S. Casado. Solving two location models with few facilities by using a hybrid heuristic: a real health resources case. *Comput. Oper. Res.*, 32(12) (2005) 3075–3091.

[31] A. P. P. Pertiwi and P. Suyanto. Globally evolved dynamic bee colony optimization. In *Knowledge-Based and Intelligent Information and Engineering Systems, Proc. 15th Int. Conf., KES 2011, Part I*, Springer Berlin Heidelberg, Kaiserslautern, Germany, (2011) 52–61.

[32] M. Sa'idi, N. Mostoufi, and R. Sotudeh-Gharebagh. Modelling and optimisation of continuous catalytic regeneration process using bee colony algorithm. *The Canadian Journal of Chemical Engineering*, 91(7) (2013) 1256–1269.

[33] M. F. Sohi, M. Shirdel, and A. Javidaneh. Applying bco algorithm to solve the optimal dg placement and sizing problem. In *5th International IEEE Power Engineering and Optimization Conference (PEOCO)*, (2011) 71–76.

[34] T. Stojanović, T. Davidović, and Z. Ognjanović. Bee-colony optimization for the satisfiability problem in probabilistic logic. *Applied Soft Computing*, 31 (2015) 339–347.

[35] D. Teodorović. 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.

[36] D. Teodorović and M. Dell'Orco. 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.

[37] D. Teodorović and M. Dell'Orco. Mitigating traffic congestion: solving the ride-matching problem by bee colony optimization. *Transport. Plan. Techn.*, 31(2) (2008) 135–152.

[38] D. Teodorović, M. Šelmić, and Lj. Mijatović-Teodorović. 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.

[39] D. Teodorović and M. Šelmić. The BCO algorithm for the *p*-median problem. In *Proceedings of the XXXIV Serbian Operations Research Conferece*, Zlatibor, Serbia (in Serbian), (2007) 417–420.

[40] M. Šelmić, P. Edara, and D. Teodorović. Bee colony optimization approach to optimize locations of traffic sensors on highways. *Tehnika* (in Serbian), 6 (2008) 9–15.

[41] M. Šelmić, D. Teodorović, and K. Vukadinović. Locating inspection facilities in traffic networks: an artificial intelligence approach. *Transport. Plan. Techn.*, 33(6) (2010) 481–493.

[42] L.-P. Wong, M. Y. Hean Low, and C. S. Chong. A bee colony optimization algorithm for traveling salesman problem. In *2-nd Asia International Conference on Modelling & Simulation*, (2008) 818–823.

[43] L.-P. Wong, M. Y. Hean Low, and C. S. Chong. Bee colony optimization with local search for traveling salesman problem. In *6-th IEEE International Conference on Industrial Informatics*, (2008) 1019–1025.

[44] L.-P. Wong, M. Y. Hean Low, and C. S. Chong. An efficient bee colony optimization algorithm for traveling salesman problem using frequency-based pruning. In *7-th IEEE International Conference on Industrial Informatics*, (2009) 775–782.

[45] L-P. Wong, M. Y. Hean Low, and C. S. Chong. Bee colony optimization with local search for traveling salesman problem. *International Journal on Artificial Intelligence Tools*, 19(03) (2010) 305–334.

[46] L. Zadeh. Fuzzy sets. *Information and Control*, 8 (1965) 338–353.