A Honey Bees Mating Optimization Algorithm for the Open Vehicle Routing Problem

Yannis Marinakis
Decision Support Systems Laboratory
Department of Production Engineering and Management
Technical University of Crete
73100 Chania, Greece
marinakis@ergasya.tuc.gr

Magdalene Marinaki
Industrial Systems Control Laboratory
Department of Production Engineering and Management
Technical University of Crete
73100 Chania, Greece
magda@dssl.tuc.gr

ABSTRACT

Honey Bees Mating Optimization algorithm is a relatively new nature inspired algorithm. In this paper, this nature inspired algorithm is used in a hybrid scheme with other metaheuristic algorithms for successfully solving the Open Vehicle Routing Problem. More precisely, the proposed algorithm for the solution of the Open Vehicle Routing Problem, the Honey Bees Mating Optimization (HBMOOVRP), combines a Honey Bees Mating Optimization (HBMO) algorithm and the Expanding Neighborhood Search (ENS) algorithm. Two set of benchmark instances is used in order to test the proposed algorithm. The results obtained for both sets are very satisfactory. More specifically, in the fourteen instances proposed by Christofides, the average quality is 0.35% when a hierarchical objective function is used, where, first, the number of vehicles is minimized and, afterwards, the total travel distance is minimized and the average quality is 0.42% when only the travel distance is minimized, while for the eight instances proposed by Li et al. when a hierarchical objective function is used the average quality is 0.21%.

TRACK: Ant Colony Optimization and Swarm Intelligence Track

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*; H.4.2 [**Information Systems Applications**]: Types of Systems—*Logistics*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Graph algorithms, Network problems, Path and circuit problems*

General Terms

Algorithms

GECCO'11, July 12–16, 2011, Dublin, Ireland.

Keywords

Honey Bees Mating Optimization, Open Vehicle Routing Problem, Expanding Neighborhood Search

1. INTRODUCTION

In the last years, several biological and natural processes have been influencing the methodologies in science and technology in an increasing manner. Among them, a number of swarm intelligence algorithms based on the behaviour of the bees has been presented [16]. These algorithms are divided, mainly, in two categories according to their behaviour in the nature, the foraging behaviour and the mating behaviour. The most known algorithm based on the marriage behaviour of the bees is the Honey Bees Mating Optimization Algorithm (HBMO) that was presented in [1, 2] and simulates the mating process of the queen of the hive. Since then, it has been used on a number of different applications [3, 9, 14, 20, 21, 22, 23, 24, 25, 35].

In this paper, as there are not any competitive nature inspired methods based to Honey Bees Mating Optimization algorithm for the solution of the Open Vehicle Routing Problem (OVRP), at least to our knowledge, we would like to propose such an algorithm and to test its efficiency compared to other nature inspired and classic metaheuristic algorithms. The proposed algorithm adopts the basic characteristics of the initially proposed Honey Bees Mating Optimization algorithm [1, 2, 3, 9, 14] and also makes a combined use of a number of different procedures in each of the subphases of the main algorithm in order to increase the efficiency of the proposed algorithm [20, 21, 22, 23, 24, 25]. The combination of all these procedures reduces, significantly, the computational time of the algorithm making the algorithm faster and more efficient and, thus, suitable for solving very large scaled problems in short computational time. The rest of the paper is organized as follows: in the next section a description of the open vehicle routing problem is presented. In section 3 the proposed algorithm, the Honey Bees Mating Optimization for the Open Vehicle Routing problem (HBMOOVRP), is presented and analyzed in detail. Computational results are presented and analyzed in section 4 while in the last section conclusions and future research are given.

2. THE OPEN VEHICLE ROUTING PROBLEM

The **Vehicle Routing Problem (VRP)** or the **capa-**

citated vehicle routing problem (CVRP) is often described as the problem in which vehicles based on a central depot are required to visit geographically dispersed customers in order to fulfill known customer demands. The vehicle routing problem was first introduced by Dantzig and Ramser (1959) [7]. Since then a number of variants of the classic Vehicle Routing Problem has been proposed in order to incorporate more constraints like time windows, multi-depot, stochastic or dynamic demand [13, 36]. The **Open Vehicle Routing Problem (OVRP)** is the variant of the classic vehicle routing problem where the vehicles do not return in the depot after the service of the customers [30]. The real life application of the Open Vehicle Routing Problem concerns the case where either the company does not have vehicles at all or the vehicles owned by the company are not enough in order to use them for the distribution of the products to the customers. In both cases the company has to hire a number of vehicles in order to realize the distribution of the products. When the vehicles finish their jobs they do not return to the company. This problem also belongs in the category of the third party logistics (3PL) problems. From the combinatorial optimization point of view, the main difference between the Vehicle Routing Problem and the Open Vehicle Routing Problem is that in the first case the route is a hamiltonian cycle while in the second case the route is a hamiltonian path [4].

The Open Vehicle Routing Problem can be stated as follows: Let $G = (V, E)$ be a graph where $V = \{j_0, j_1, j_2, \cdots j_n\}$ is the vertex set ($j_i = j_0$ refers to the depot and the customers are indexed $j_i = j_1, \cdots, j_n$) and $E = \{(j_l, j_{l_1}) : j_l, j_{l_1} \in V\}$ is the edge set. Each customer must be assigned to exactly one of the k vehicles and the total size of deliveries for customers assigned to each vehicle must not exceed the vehicle capacity (Q_k). If the vehicles are homogeneous, the capacity for all vehicles is equal and denoted by Q. Each vehicle has the same traveling cost L. A demand q_{j_l} and a service time st_{j_l} are associated with each customer node j_l. The travel cost between customers j_l and j_{l_1} is $cost_{j_l j_{l_1}}$. The problem is to construct a low cost, feasible set of routes - one for each vehicle. A route is a sequence of locations that a vehicle must visit along with the indication of the service it provides. Each vehicle starts at the depot but it doesn't return to the depot. The total traveling cost of each route can not exceed the restriction L. Usually two different objectives are used in OVRP, the first one is the minimization of the required number of vehicles and the second one is the minimization of the corresponding total traveled distance.

The Open Vehicle Routing Problem is an NP-hard problem. The instances with a large number of customers cannot be solved in optimality within reasonable time. For this reason a large number of approximation techniques has been proposed for its solution. These techniques are classified into three main categories: the classical heuristics, the single solution based metaheuristics and the population based metaheuristics. The Open Vehicle Routing Problem was first published in [31] but since then for the following twenty years it received little study. In the last ten years, a number of publications using different heuristic and metaheuristic algorithms for the OVRP have been published. More precisely, algorithms based on classic heuristics [30], tabu search [4, 8, 11], record to record travel [17], adaptive memory [32], backtracking adaptive threshold accepting [33], list based threshold accepted algorithm [34], adaptive large neighborhood search [27], differential evolution [5], evolution strategy [28], variable neighborhood search [10], broad local search algorithm [38], ant colony optimization [18, 19] and particle swarm optimization [37] have been proposed in order to give efficient alternative algorithms for the solution of the OVRP.

3. HONEY BEES MATING OPTIMIZATION FOR THE OPEN VEHICLE ROUTING PROBLEM

The proposed algorithm, the **Honey Bees Mating Optimization Algorithm for the Open Vehicle Routing Problem** (**HBMOOVRP**), combines a number of different procedures. Each of them corresponds to a different phase of the mating process of the honey bees.

One of the key issues in designing a successful HBMO algorithm for the Open Vehicle Routing Problem is to find a suitable mapping between Open Vehicle Routing Problem solutions and bees in HBMO. Each bee is recorded via the path representation of the tour, that is, via the specific sequence of the nodes. For example if we have a bee (solution) with ten nodes, a possible path representation is the following:

1 3 8 5 4 10 1 6 9 7 2

with node number 1 is denoted the depot and nodes 2 through 10 denote the customers. The difference between the Open Vehicle Routing Problem and the Capacitated Vehicle Routing Problem is that in the first the vehicles do not return to the depot. Thus, the difference in the calculation of a cost function for each bee is that we do not add the cost between the last customer and the depot, i.e. in the previous example the cost (distances) between customer 10 and depot and customer 2 and the depot are omitted.

Initially, we have to choose the population of the honey bees that will configure the initial hive. There are two different ways to calculate the initial population, either completely at random (as in the initially proposed algorithm [1]) or by using an algorithm in order to obtain as good initial solutions as possible, and, thus, to obtain a more efficient queen [21]. In the proposed algorithm, the initial population is created at random in order to give to the proposed algorithm the abilities to search in the whole solution space and not to restrict the solution near to some good initial solutions. The best member of the initial population of bees is selected as the queen of the hive. All the other members of the population are the drones.

Before the process of mating begins, the user has to define a number that corresponds to the queen's size of spermatheca. This number corresponds to the maximum number of queen's mating in a single mating flight. Each time the queen successfully mates with a drone the genotype of the drone is stored and a variable is increased by one until the size of spermatheca is reached [2]. Another two parameters have to be defined, the number of queens and the number of broods that will be born by all queens. In this implementation of Honey Bees Mating Optimization (HBMO) Algorithm, the number of queens is set equal to one as in the real life only one queen will survive in a hive, and the number of broods is set equal to the number corresponding to the size of queen's spermatheca. Then, the mating flight of the queen begins. At the start of the flight, the queen is initialized with some energy content (initially, the speed and the energy of the queen are generated at random) and

returns to her nest when the energy is less than a threshold value ($thres$) and spermatheca is not full [3]. A drone mates with a queen probabilistically using the following annealing function [1, 2]:

$$Prob(D) = e^{[\frac{-\Delta(f)}{Speed(t)}]} \quad (1)$$

where $Prob(D)$ is the probability of adding the sperm of drone D to the spermatheca of the queen (that is, the probability of a successful mating), $\Delta(f)$ is the absolute difference between the fitness of D and the fitness of the queen (for complete description of the calculation of the fitness function see below) and $Speed(t)$ is the speed of the queen at time t. The probability of mating is high when the queen is at the beginning of her mating flight, therefore her speed is high, or when the fitness of the drone is as good as the queen's. After each transition in space, the queen's speed and energy decays according to the following equations:

$$Speed(t+1) = \alpha \times Speed(t) \quad (2)$$

$$energy(t+1) = \alpha \times energy(t) \quad (3)$$

where α is a factor $\in (0, 1)$ that determines the amount that the speed and the energy will be reduced after each transition and each step. It should be noted that equation (3) is different than the one proposed by [1, 2] and it was introduced in this form as we would like to straightforwardly correlate the reduction of the speed with the reduction of the energy and, also, to use as less as possible parameters. Initially, the speed and the energy of the queen are generated at random. A number of mating flights are realized. At the start of a mating flight drones are generated randomly and the queen selects a drone using the probabilistic rule in Eq. (1). If the mating is successful (i.e., the drone passes the probabilistic decision rule), the drone's sperm is stored in the queen's spermatheca.

By using a crossover operator a new brood (trial solution) is formed which later can be improved, employing workers to conduct local search. The crossover operator proposed in [21] is used in order to simulate the procedure that occurs in real life where the queen stores a number of different drone's sperm in her spermatheca and uses parts of the genotype of the different drones to create the new brood. Thus, the quality of the new solution (the brood) is fittest because as it takes parts of different solutions (queen and drones) it has more exploration abilities. This is a major difference of the proposed algorithm compared to other honey bees mating optimization algorithms [1, 3, 9, 14, 35] and to the classic evolutionary algorithms.

In real life, the role of the workers is restricted to brood care and for this reason the workers are not separate members of the population but they are used as local search procedures in order to improve the broods produced by the mating flight of the queen. Each of the workers have different capabilities and the choice of two different workers may produce different solutions. This is realized with the use of a number of single local search heuristics (w_1) and combinations of them (w_2). Thus, the sum of this two numbers ($w = w_1 + w_2$) gives the number of workers. Each of the brood will choose, randomly, one worker to feed it (local search phase) having as a result the possibility of replacing the queen if the solution of the brood is better than the solution of the current queen. If the brood fails to replace the queen, then in the next mating flight of the queen this brood will be one of the drones. A pseudocode of the proposed algorithm is presented in the following while in the next paragraphs some procedures of the algorithm are explained in detail.

algorithm `Honey Bees Mating Optimization for OVRP`
Initialization
Generate the initial population of the bees
Selection of the best bee as the queen
Select maximum number of mating flights (M)
do while $i \leq M$
    Initialize queen's spermatheca, energy and speed.
    Select α
    do while $energy > thres$ and $spermatheca\ is\ not\ full$
        Select a drone
        if the drone passes the probabilistic condition
            Add sperm of the drone in the spermatheca
        endif
    $Speed(t+1) = \alpha \times Speed(t)$
    $energy(t+1) = \alpha \times energy(t)$
    enddo
    do $j = 1, Size\ of\ Spermatheca$
        Select a sperm from the spermatheca
        Generate a brood by applying a crossover
            operator between the queen, the selected
            drones and the adaptive memory
        Select, randomly, a worker
        Use the selected worker to improve
            the brood's fitness
        if the brood's fitness is better than
            the queen's fitness **then**
            Replace the queen with the brood
        else
            Add the brood to the population of drones
        endif
    enddo
enddo
return The Queen (Best Solution Found)

3.1 Calculation of Fitness Function

In OVRP, the fitness of each individual is related to the route length of each circle. Since the problems that we deal with are minimization problems, if a feasible solution has a large objective function value then it is characterized as an unpromising solution candidate and, therefore, its fitness must be set to a small value. Reversely, a large fitness value must correspond to a solution with a low objective function value. A way to accomplish this, is to find initially the individual in the population with the maximum objective function value and to subtract from this value the objective function value of each of the other individuals. By doing this, the larger fitness value corresponds to the tour with the shorter length. Since the probability of selecting an individual for mating is related to its fitness and since the individual with the worst objective function value has fitness equal to zero, it will never be selected for mating. Therefore, in order to avoid its total exclusion, the fitness of all individuals in this population is incremented by one, resulting, thus in a worse individual of fitness one.

3.2 Crossover Operator

We use a multiparent crossover operator which initially identifies the common characteristics of the parent individuals and, then, copies them to the broods. This crossover operator is a kind of adaptive memory procedure. Initially, the adaptive memory has been proposed by Rochat and Taillard [29] as a part of a tabu search metaheuristic for the solution of the Vehicle Routing Problem. This procedure stores characteristics (partial or complete solutions in the Open Vehicle Routing Problem) of good solutions. Each time a new good solution has been found the adaptive memory is updated. In our case, in the first generation the adaptive memory is empty. In order to add a solution or a part of a solution in the adaptive memory there are three possibilities:

1. The candidate for the adaptive memory solution is a previous best solution (queen) that has fitness function value at most 10% worse than the value of the current best solution.
2. The candidate for the adaptive memory solution is a member of the population (drone) that has fitness function value at most 10% worse than the value of the current best solution.
3. A partial solution (i.e. a path) is common for the queen and for a number of drones.

More analytically, in this crossover operator, the points are selected randomly from the adaptive memory, from the selected drones and from the queen. Thus, initially two crossover operator numbers are selected (Cr_1 and Cr_2) that control the fraction of the parameters that are selected for the adaptive memory, the selected drones and the queen. If there are common parts in the solutions (queen, drones and adaptive memory) then these common parts are inherited to the brood, else the Cr_1 and Cr_2 values are compared with the output of a random number generator, $rand_i(0,1)$. If the random number is less or equal to the Cr_1 the corresponding value is inherited from the queen, if the random number is between the Cr_1 and the Cr_2 then the corresponding value is inherited, randomly, from the one of the elite solutions that are in the adaptive memory, otherwise it is selected, randomly, from the solutions of the drones that are stored in spermatheca. Thus, if the solution of the queen is denoted by $q_i(t)$ (t is the iteration number), the solution in the adaptive memory is denoted by $ad_i(t)$ and the solution of the drone by $d_i(t)$, then, the solution of the brood $b_i(t)$ is given by:

$$b_i(t) = \begin{cases} q_i(t), & \text{if } rand_i(0,1) \leq Cr_1 \\ ad_i(t), & \text{if } Cr_1 < rand_i(0,1) \leq Cr_2 \\ d_i(t), & \text{otherwise.} \end{cases} \quad (4)$$

In each iteration, the adaptive memory is updated based on the best solution.

3.3 Workers - Expanding Neighborhood Search

As it has already been mentioned, the workers are not separate members of the population but they are used as local search procedures in order to improve the broods produced by the mating flight of the queen. The local search method that is used in this paper is the Expanding Neighborhood Search [26]. Expanding Neighborhood Search (ENS) is a metaheuristic algorithm [26] that can be used for the solution of a number of combinatorial optimization problems with remarkable results. The main features of this algorithm are:

- the use of the Circle Restricted Local Search Moves Strategy,
- the ability of the algorithm to change between different local search strategies, and,
- the use of an expanding strategy.

These features are explained in detail in the following.

In the Circle Restricted Local Search Moves (CRLSM) strategy, the computational time is decreased significantly compared to other heuristic and metaheuristic algorithms because all the edges that are not going to improve the solution are excluded from the search procedure. This happens by restricting the search space into circles around the candidate for deletion edges. It has been observed [26], for example, in the 2-opt local search algorithm that there is only one possibility for a trial move to reduce the cost of a solution, i.e. when at least one new (candidate for inclusion) edge has cost less than the cost of one of the two old edges (candidate for deletion edges) and the other edge has cost less than the sum of the costs of the two old edges. Thus, in the Circle Restricted Local Search Moves strategy, for all selected local search strategies, circles are created around the end nodes of the candidate for deletion edges and only the nodes that are inside these circles are used in the process of finding a better solution.

In order to decrease even more the computational time and because it is more possible to find a better solution near to the end-nodes of the candidate for deletion edge, we do not use from the begin the largest possible circle but the search for a better solution begins with a circle with a small radius. For example, in the 2-opt algorithm if the length of the candidate for deletion edge is equal to A, the initial circle has radius $A/2$, then, the local search strategies are applied as they are described in the following and if the solution can not be improved inside this circle, the circle is expanding by a percentage θ (θ is determined empirically) and the procedure continues until the circle reaches the maximum possible radius which is set equal to $A + B$, where B is the length of one of the other candidate for deletion edges.

The ENS algorithm has the ability to change between different local search strategies. The idea of using a larger neighborhood to escape from a local minimum to a better one, had been proposed initially by Garfinkel and Nemhauser [12] and recently by Hansen and Mladenovic [15]. Garfinkel and Nemhauser proposed a very simple way to use a larger neighborhood. In general, if with the use of one neighborhood a local optimum was found, then a larger neighborhood is used in an attempt to escape from the local optimum. Hansen and Mladenovic proposed a more systematical method to change between different neighborhoods, called Variable Neighborhood Search.

In the Expanding Neighborhood Search a number of local search strategies are applied inside the circle. The procedure works as follows: initially an edge of the current solution is selected (for example the edge with the worst length) and the first local search strategy is applied. If with this local search strategy a better solution is not achieved, another

local search strategy is selected for the same edge. This procedure is continued until a better solution is found or all local search strategies have been used. In the first case the solution is updated, a new edge is selected and the new iteration of the Expanding Neighborhood Search strategy begins, while in the second case the circle is expanded and the local search strategies are applied in the new circle until a better solution is found or the circle reach the maximum possible radius. If the maximum possible radius has been reached, then a new candidate for deletion edge is selected.

4. COMPUTATIONAL RESULTS

The algorithm was implemented in Fortran 90 and was compiled using the Lahey f95 compiler on a Intel Core 2 DUO CPU T9550 at 2.66 GHz, running Suse Linux 9.1.

The algorithm was tested on two sets of benchmark problems, the 14 benchmark problems proposed by Christofides [6] and the 8 large scale open vehicle routing problems proposed by Li et al. [17]. Each instance of the first set contains between 51 and 200 nodes including the depot. The location of the nodes is defined by their Cartesian co-ordinates and the travel cost from node i to j is assumed to be the respective Euclidean distance. Each problem includes capacity constraints while the problems 6-10, 13 and 14 have, also, maximum route length restrictions (mtl) and non zero service times (st). For the first ten problems, nodes are randomly located over a square, while for the remaining ones, nodes are distributed in clusters and the depot is not centred. The maximum allowed route length has been multiplied by 0.9 compared to the one considered for the VRP [28]. The second set of instances contains between 200 and 480 nodes including the depot. Each problem instance includes capacity constraints. In Tables 1 and 2 the most important characteristics of each of the data set are presented.

Table 1: Benchmark instances of [6]

number	n	Capacity	mtl	st
C1	51	160	∞	0
C2	76	140	∞	0
C3	101	200	∞	0
C4	151	200	∞	0
C5	200	200	∞	0
C6	51	160	180	10
C7	76	140	144	10
C8	101	200	207	10
C9	151	200	180	10
C10	200	200	180	10
C11	121	200	∞	0
C12	101	200	∞	0
C13	121	200	648	50
C14	101	200	936	90

Table 2: Benchmark instances of [17]

number	n	Capacity
O1	200	900
O2	240	550
O3	280	900
O4	320	700
O5	360	900
O6	400	900
O7	440	900
O8	480	1000

The parameters of the proposed algorithm are selected after thorough testing. A number of different alternative values were tested and the ones selected are those that gave the best computational results concerning both the quality of the solution and the computational time needed to achieve this solution. Thus, the selected parameters are:

- number of queens equal to 1,
- number of drones equal to 200,
- number of mating flights (M) equal to 1000,
- size of queen's spermatheca equal to 50,
- number of broods equal to 50,
- α equal to 0.9,
- θ equal to 10%,
- number of workers (w) equal to 20, ($w_1 = 7, w_2 = 13$).

More precisely, the reason that we chose the number of queens equal to 1 is that in real life only one queen is in a hive. The mating flights (the iterations of the algorithm) is set equal to 1000 as we would like to keep the executable time less than 5 minutes even in the more difficult example which is the instance O8 with number of nodes equal to 480 and, thus, keeping the iterations equal to 1000 the algorithm did not exceed the limitation of five minutes in any instance. The reason that the queen's spermatheca is set equal to 50 is that we would like to use a percentage of the drones in the crossover phase of the algorithm and, thus, we select to use at most 25% of the drones. We test the algorithm with five different sizes of the spermatheca (5%, 10%, 25%, 50%, 75% of the drones). We observed that in the first two cases a number of good drones were not selected and thus the algorithm needed more iterations to converge. When the size of the spermatheca was set equal to 75% of the drones the algorithm became slower and the gain in the convergence and in the quality was not important. The reason that the number of broods was set equal to 50 is that, as in the end of the iterations the broods will replace the drones (or a number of drones) in order to keep the number of population stable in each iteration, a ration between the drones and the broods equal to 25% gave to the algorithm the possibility of keeping good drones in a large number of generations. The reason that the number of drones is set equal to 200 is that as we would like to increase the exploration and the exploitation abilities of the algorithm, a large number of drones helped the algorithm to search in the most of the search space. A number of nodes equal to 100 it will be enough, but then we have to change the ratio of the drones with the broods and the size of spermatheca, and as the use of 200 drones did not increase significantly the computational time of the algorithm (the time needed to solve the most difficult example was never more than five minutes), we selected the drones equal to 200. There are two reasons that the number of workers was set more than one. First, one of the characteristics of the Expanding Neighborhood Search is that it uses in the local search phase more than one local search procedures in order to give the opportunity to the algorithm to have more exploitation abilities. Second, in the HBMO algorithm a local search works as a brood care, meaning a good local search algorithm will help

to have a fittest brood. As a local search algorithm will perform differently for each solution we gave the possibility to each brood to select randomly a different worker (single or combination of local search phases). The local search phases that are used in the algorithm are the 2-opt, 3-opt, 1-0 relocate, 2-0 relocate, 1-1 exchange, 2-2 exchange, the crossing algorithm and 13 different combinations of them.

The efficiency of the HBMOOVRP algorithm is measured by the quality of the produced solutions. The quality is given in terms of the relative deviation from the best known solution, that is $\omega = \frac{(c_{HBMOOVRP} - c_{BKS})}{c_{BKS}}\%$, where $c_{HBMOOVRP}$ denotes the cost of the solution found by HBMOOVRP and c_{BKS} is the cost of the best known solution.

The results of the proposed algorithm for the first data set are presented in Table 3. To test the performance of the proposed algorithm we applied HBMOOVRP 10 times to each test problem. In Table 3 the best results, the average results, the median, the standard deviation (stdev) and variance (var) are presented. In this Table two different best solutions are presented. The first best known solution (BKS1) is obtained using first the minimization of the number of vehicles and then the minimization of the total distance traveled. The other best solution (BKS2) is obtained by minimizing only the total distance traveled. As we explained in the description of the problem it is very important to use the smallest number of vehicles as in the real life application of the Open Vehicle Routing Problem the finding of the best routes by hiring as less as possible number of vehicles is the main concern. Thus, initially we solved the Open Vehicle Routing Problem with the proposed algorithm using the hierarchical objective function, where initially the number of vehicles is minimized and, then, for this number of vehicles the total travel distance is, also, minimized. The results, the quality of the solution, the average results, the median, the standard deviation (stdev) and variance (var) and the CPU time in minutes, are presented in the first part of the Table 3 and in columns 4 to 10. Afterwards, we solved the Open Vehicle Routing Problem with the single objective function (the total distance traveled). The results, the quality of the solution, the average results, the median, the standard deviation (stdev), the variance (var) and the CPU time in minutes, are presented in the second part of the Table 3 and in columns 4 to 10. For the second case, there are a number of instances that have no values in Table 3. The reason is that in these instances the results are the same as the results obtained for the first case.

It can be seen from Table 3, that the HBMOOVRP algorithm, in five out of fourteen instances in the first case and in four out of fourteen instances (together with the two which their solutions are the same as in the first case) has reached the best known solution. For the rest instances in the first case (when a hierarchical objective function is used) the quality of the solutions is between 0.08% and 1.07% and the average quality for the fourteen instances is 0.35%. For the second case (when only the travel distance is minimized) the quality of the solutions is between 0.13% and 1.19% and the average quality for the fourteen instances is 0.42%. The standard deviation in the first case is between 0.29 and 0.51 while in the second case is between 0.20 and 0.50. The variance in the first case is between 0.08 and 0.26 while in the second case is between 0.04 and 0.25. Also, in this Table the computational time needed (in minutes) for finding the best solution by HBMOOVRP is presented. The CPU time needed is significantly low and only for the instances with number of nodes equal to 200 is larger than 3 minutes.

The algorithm is also tested for the large scale benchmark instances proposed by Li et al [17]. The reuslts of the second data set are presented in Table 4. In this data set, we present only results for the hierarchical objective function where first the number of vehicles is minimized and afterwards the total distance traveled is minimized. The results, the quality of the solution, the average results, the median, the standard deviation (stdev) and variance (var) and the CPU time in minutes are presented in the first part of the Table 4 and in columns 4 to 10. The quality of the solutions for the 8 instances is between 0.08% and 0.52% and the average quality is 0.21%. The standard deviation is between 0.30 and 0.53 and the variance is between 0.09 and 0.28. Also, in this Table the computational time needed (in minutes) for finding the best solution by HBMOOVRP is presented. The CPU time needed is significantly low and never is larger than 5 minutes.

5. CONCLUSIONS AND FUTURE RESEARCH

In this paper, a nature inspired approach was introduced for the effective handling of the Open Vehicle Routing Problem (OVRP). More specifically, a hybrid algorithmic nature inspired methodology was proposed, namely the Honey Bees Mating Optimization algorithm for the OVRP (HBMOOVRP) that gave remarkable results both to quality and computational efficiency. The algorithm was applied in a set of benchmark instances and gave very satisfactory results. More specifically, in the set with the classic benchmark instances proposed by Christofides, the average quality is 0.35% when a hierarchical objective function is used, where first the number of vehicles is minimized and, then, for this number of vehicles the total travel distance is minimized and the average quality is 0.42% when only the total travel distance is minimized. For the large scale instances proposed by Li et al., when a hierarchical objective function is used, the average quality is 0.21%. In the future, we would like to test the performance of the algorithm in other variants of the Vehicle Routing Problem like the Vehicle Routing Problem with Time Windows and the Multiple Depot Vehicle Routing Problem.

6. REFERENCES

[1] Abbass, H.A.: A monogenous MBO approach to satisfiability. In: Proceeding of the International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA'2001, Las Vegas, NV, USA, (2001)

[2] Abbass, H.A.: Marriage in honey-bee optimization (MBO): A haplometrosis polygynous swarming approach. The Congress on Evolutionary Computation (CEC2001), Seoul, Korea, May 2001, 207-214 (2001)

[3] Afshar, A., Bozog Haddad, O., Marino, M.A., Adams, B.J.: Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation. Journal of the Franklin Institute 344, 452-462 (2007)

[4] Brandao, J.: A tabu search algorithm for the open vehicle routing problem. European Journal of Operational Research 157(3), 552-564 (2004)

Table 3: Results of HBMOOVRP in Christofides benchmark instances

Results with hierarchical objective function									
	BKS1	NV	HBMOOVRP	ω	average	median	stdev	var	CPU (min)
C1	416.06[4]	5	416.06	0.00	416.46	416.39	0.43	0.18	0.09
C2	567.14 [11]	10	567.14	0.00	567.49	567.39	0.38	0.15	0.35
C3	639.74 [17]	8	640.25	0.08	640.66	640.48	0.41	0.17	0.58
C4	733.13 [27]	12	738.49	0.73	738.97	738.92	0.37	0.14	2.17
C5	893.39 [38]	16	902.17	0.98	902.78	902.76	0.48	0.23	3.18
C6	412.96 [4]	6	412.96	0.00	413.38	413.48	0.32	0.10	0.11
C7	583.19 [27]	10	583.19	0.00	583.64	583.58	0.37	0.14	0.25
C8	644.63 [4]	9	644.63	0.00	645.14	645.21	0.42	0.18	0.48
C9	757.84 [27]	13	765.95	1.07	766.33	766.18	0.40	0.16	1.57
C10	875.67 [27]	17	884.28	0.98	884.77	884.68	0.48	0.23	3.35
C11	682.12 [27]	7	683.15	0.15	683.70	683.76	0.51	0.26	1.05
C12	534.24 [27]	10	536.37	0.40	536.72	536.74	0.29	0.08	1.28
C13	904.04 [10]	11	905.18	0.13	905.78	905.82	0.34	0.11	1.23
C14	591.87 [27]	11	593.95	0.35	594.37	594.23	0.35	0.12	1.42
Results with minimizing the total distance									
	BKS2	NV	HBMOOVRP	ω	average	median	stdev	var	CPU (min)
C1	412.96 [32]	6	412.96	0.00	413.29	413.19	0.36	0.13	0.17
C2	564.06 [32]	11	564.06	0.00	564.34	564.37	0.20	0.04	0.42
C3	639.26 [38]	9	640.08	0.13	640.51	640.42	0.40	0.16	0.55
C5	869 [38]	17	878.25	1.06	878.79	878.68	0.50	0.25	3.05
C7	568.49 [17]	11	575.25	1.19	575.84	575.84	0.50	0.25	0.37
C9	756.38 [17]	14	761.41	0.67	761.85	761.76	0.42	0.17	1.15
C11	678.54 [34]	10	680.15	0.24	680.51	680.42	0.36	0.13	1.17
C13	896.5 [17]	12	898.18	0.19	898.50	898.49	0.28	0.08	1.35

Table 4: Results of HBMOOVRP in Li et al. benchmark instances

Results with hierarchical objective function									
	BKS1	NV	HBMOOVRP	ω	average	median	stdev	var	CPU (min)
O1	6018.52 [17]	5	6023.48	0.08	6023.75	6023.66	0.30	0.09	2.58
O2	4557.38 [38]	9	4561.18	0.08	4561.68	4561.68	0.40	0.16	3.12
O3	7731 [38]	7	7745.16	0.18	7745.80	7745.86	0.31	0.10	3.25
O4	7253.2 [38]	10	7287.49	0.47	7287.92	7287.86	0.42	0.18	3.42
O5	9193.15 [38]	8	9201.25	0.09	9201.59	9201.48	0.31	0.09	3.57
O6	9793.72 [38]	9	9809.48	0.16	9810.03	9809.9	0.53	0.28	4.15
O7	10347.7 [38]	10	10401.24	0.52	10401.65	10401.60	0.34	0.11	4.38
O8	12415.36 [38]	10	12429.57	0.11	12430.03	12429.92	0.37	0.14	4.59

[5] Cao, E., Lai, M.: The open vehicle routing problem with fuzzy demands. Expert Systems with Applications 37, 2405-2411 (2010)

[6] Christofides, N., Mingozzi, A., Toth, P.: The vehicle routing problem. In: Christofides, N., Mingozzi, A., Toth, P., Sandi, C. (eds.), Combinatorial Optimization, Wiley, Chichester, (1979)

[7] Dantzig, G.B., Ramser, J.H.: The truck dispatching problem. Management Science 6(1), 80-91 (1959)

[8] Derigs, U., Reuter, K.: A simple and efficient tabu search heuristic for solving the open vehicle routing problem. Journal of Operational Research Society 60, 1658-1669 (2009)

[9] Fathian, M., Amiri, B., Maroosi, A.: Application of honey bee mating optimization algorithm on clustering. Applied Mathematics and Computation 190, 1502-1513 (2007)

[10] Fleszar, K., Osman, I.H., Hindi, K.S.: A variable neighbourhood search algorithm for the open vehicle routing problem. European Journal of Operational Research 195, 803-809 (2009)

[11] Fu, Z., Eglese, R., Li, L.: A new tabu search heuristic for the open vehicle routing problem. Journal of the Opeartional Research Society 56(2), 267-274 (2005)

[12] Garfinkel R., Nemhauser G.: Integer programming. Wiley and Sons, New York, (1972)

[13] Golden, B.L., Assad. A.A. Vehicle Routing: Methods and Studies. North Holland, Amsterdam, (1988)

[14] Haddad, O.B., Afshar, A., Marino, M.A.: Honey-bees mating optimization (HBMO) algorithm: A new heuristic approach for water resources optimization. Water Resources Management 20, 661-680 (2006)

[15] Hansen, P., Mladenovic, N. Variable neighborhood search: Principles and applications. European Journal of Operational Research 130, 449-467 (2001)

[16] Karaboga, D., Akay, B.: A survey: Algorithms simulating bee swarm intelligence. Artificial Intelligence Review 31, 61-85 (2009)

[17] Li, F., Golden, B., Wasil, E.: The open vehicle routing problem: Algorithms, large-scale test problems, and computational results. Computers and Operations Research 34, 2918-2930 (2007)

[18] Li, X., Tian, P.: An ant colony system for the open vehicle routing problem. In: Dorigo, M., et al. (eds.), ANTS 2006, LNCS 4150, 356-363 (2006)

[19] Li, X.Y., Tian, P., Leung, S.C.H.: An ant colony optimization metaheuristic hybridized with tabu

search for open vehicle routing problems. Journal of Operational Research Society 60(7), 1012-1025 (2009)
[20] Marinaki, M., Marinakis, Y., Zopounidis, C.: Honey bees mating optimization algorithm for financial classification problems. Applied Soft Computing 10, 806-812 (2010)
[21] Marinakis Y., Marinaki, M.: A hybrid honey bees mating optimization algorithm for the probabilistic traveling salesman problem. IEEE Congress on Evolutionary Computation (CEC 2009), Trondheim, Norway, (2009)
[22] Marinakis, Y., Marinaki, M., Dounias, G.: Honey bees mating optimization algorithm for the vehicle routing problem. In: Krasnogor, N., Nicosia, G., Pavone, M., Pelta, D. (eds.) Nature Inspired Cooperative Strategies for Optimization - NICSO 2007, Studies in Computational Intelligence, Springer-Verlag Berlin 129, 139-148 (2008)
[23] Marinakis, Y., Marinaki, M., Dounias, G.: Honey bees mating optimization algorithm for large scale vehicle routing problems. Natural Computing 9, 5-27 (2010)
[24] Marinakis, Y., Marinaki, M., Dounias, G. Honey bees mating optimization algorithm for the euclidean traveling salesman problem. Information Sciences (available on line - DOI: 10.1016/j.ins.2010.06.032) (2010)
[25] Marinakis, Y., Marinaki, M., Matsatsinis, N. Honey bees mating optimization for the location routing problem. IEEE International Engineering Management Conference (IEMC - Europe 2008), Estoril, Portugal, (2008)
[26] Marinakis, Y., Migdalas, A., Pardalos, P.M.: Expanding neighborhood GRASP for the traveling salesman problem. Computational Optimization and Applications 32, 231-257 (2005)
[27] Pisinger, D., Ropke, S.: A general heuristic for vehicle routing problems. Computers and Operations Research 34, 2403-2435 (2006)
[28] Repoussis, P.P., Tarantilis, C.D., Braysy, O., Ioannou, G.: A hybrid evolution strategy for the open vehicle routing problem. Computers and Operations Research 37, 443-455 (2010)
[29] Rochat, Y., Taillard, E.D.: Probabilistic diversification and intensification in local search for vehicle routing. Journal of Heuristics 1, 147-167 (1995)
[30] Sariklis, D., Powell, S.: A heuristic method for the open vehicle routing problem. The Journal of the Operational Research Society 51(5), 564-573 (2000)
[31] Schrage, L.: Formulation and structure of more complex realistic routing and scheduling problem. Networks 11, 229-232 (1981)
[32] Tarantilis, C., Diakoulaki, D., Kiranoudis, C.: Combination of geographical information system and efficient routing algorithms for real life distribution operations. European Journal of Opeartions Research 152(2), 437-453 (2004)
[33] Tarantilis, C., Ioannou, G., Kiranoudis, C., Prastacos, G.: A threshold accepting approach to the open vehicle routing problem. RAIRO Operations Research 38, 345-360 (2004)
[34] Tarantilis, C., Ioannou, G., Kiranoudis, C., Prastacos, G.: Solving the open vehicle routing problem via a single parameter metaheuristic algorithm. Journal of the Operational Research Society 56, 588-596 (2005)
[35] Teo, J., Abbass, H.A.: A true annealing approach to the marriage in honey bees optimization algorithm. International Journal of Computational Intelligence and Applications 3(2), 199-211 (2003)
[36] Toth, P., Vigo, D.: The vehicle routing problem. Monographs on Discrete Mathematics and Applications, Siam, (2002)
[37] Wang, W., Wu, B., Zhao, Y., Feng, D.: Particle swarm optimization for open vehicle routing problem. In: Huang, D.-S., Li, K., Irwin, G.W. (eds.), ICIC 2006, LNAI 4114, 999-1007 (2006)
[38] Zachariadis, E.E., Kiranoudis, C.T.: An open vehicle routing problem metaheuristic for examining wide solution neighborhoods. Computers and Operations Research 37, 712-723 (2010)