Nat Comput (2010) 9:5–27
DOI 10.1007/s11047-009-9136-x

Honey Bees Mating Optimization algorithm for large scale vehicle routing problems

Yannis Marinakis · Magdalene Marinaki · Georgios Dounias

Published online: 25 June 2009

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 Vehicle Routing Problem. More precisely, the proposed algorithm for the solution of the Vehicle Routing Problem, the Honey Bees Mating Optimization (HBMOVRP), combines a Honey Bees Mating Optimization (HBMO) algorithm with the Multiple Phase Neighborhood Search–Greedy Randomized Adaptive Search Procedure (MPNS–GRASP) and the Expanding Neighborhood Search (ENS) algorithm. Besides these two procedures, the proposed algorithm has, also, two additional main innovative features compared to other Honey Bees Mating Optimization algorithms concerning the crossover operator and the workers. Two sets of benchmark instances are used in order to test the proposed algorithm. The results obtained for both sets are very satisfactory. More specifically, in the fourteen classic instances proposed by Christofides, the average quality is 0.029% and in the second set with the twenty large scale vehicle routing problems the average quality is 0.40%.

Keywords Vehicle Routing Problem · Honey Bees Mating Optimization · Multiple Phase Neighborhood Search–GRASP

Y. Marinakis (✉)
Decision Support Systems Laboratory, Department of Production Engineering and Management, Technical University of Crete, University Campus, 73100 Chania, Crete, Greece
e-mail: marinakis@ergasya.tuc.gr

M. Marinaki
Industrial Systems Control Laboratory, Department of Production Engineering and Management, Technical University of Crete, University Campus, 73100 Chania, Crete, Greece
e-mail: magda@dssl.tuc.gr

G. Dounias
Department of Financial and Management Engineering, Management and Decision Engineering Laboratory, University of the Aegean, 31 Fostini Str., 82100 Chios, Greece
e-mail: g.dounias@aegean.gr

1 Introduction

During the last decade nature inspired intelligence has become increasingly popular through the development and utilization of intelligent paradigms in advanced information systems design. Among the most popular nature inspired approaches, when the task is optimization within complex domains of data or information, are those methods representing successful animal and micro-organism team behaviour, such as swarm or flocking intelligence [birds flocks or fish schools inspired Particle Swarm Optimization (Kennedy and Eberhart 1995)], artificial immune systems (that mimic the biological one), ant colonies [ants foraging behaviors gave rise to Ant Colony Optimization (Dorigo and Stutzle 2004)], or optimized performance of bees, etc.

In the recent few years a number of swarm intelligence algorithms, based on the behaviour of the bees have been presented (Baykasoglu et al. 2007). These algorithms are divided, mainly, in two categories according to their behaviour in the nature, the foraging behaviour and the mating behaviour. The most important approaches that simulate the foraging behaviour of the bees are the Artificial Bee Colony (ABC) algorithm proposed by Karaboga and Basturk (2007, 2008), the Virtual Bee Algorithm proposed by Yang (2005), the Bee Colony Optimization algorithm proposed by Teodorovic and Dell'Orco (2005), the BeeHive algorithm proposed by Wedde et al. (2004), the Bee Swarm Optimization algorithm proposed by Drias et al. (2005) and the Bees Algorithm proposed by Pham et al. (2006). The Artificial Bee Colony algorithm (Karaboga and Basturk 2007, 2008) is, mainly, applied in continuous optimization problems and simulates the waggled dance behaviour that a swarm of bees performs during the foraging process of the bees. In this algorithm there are three groups of bees, the employed bees (bees that determines the food source (possible solutions) from a prespecified set of food sources and share this information (waggle dance) with the other bees in the hive), the onlookers bees (which are bees that based on the information taken from the employed bees they search for a better food source in the neighborhood of the memorized food sources) and the scout bees (which are employed bees that their food source has been abandoned and they search for a new food source randomly). The Virtual Bee Algorithm (Yang 2005) is, also, applied in continuous optimization problems. In this algorithm, the population of bees is associated with a memory, a food source, and then all the memories communicate between them with a waggle dance procedure. The whole procedure is similar to a genetic algorithm and it has been applied on two function optimization problems with two parameters. In the BeeHive (Wedde et al. 2004) algorithm, a protocol inspired from dance language and foraging behaviour of honey bees is used. In the Bees Swarm Optimization (Drias et al. 2005), initially a bee finds an initial solution (food source) and from this solution the other solutions are produced with certain strategies. Then, every bee is assigned in a solution and when they accomplished their search, the bees communicate between them with a waggle dance strategy and the best solution will become the new reference solution. To avoid cycling the authors use a tabu list. In the Bees Algorithm (Pham 2006), a population of initial solutions (food sources) are randomly generated. Then, the bees are assigned to the solutions based on their fitness function. The bees return to the hive and based on their food sources, a number of bees are assigned to the same food source in order to find a better neighborhood solution. In the Bee Colony Optimization (Teodorovic and Dell'Orco 2005) algorithm, a step by step solution is produced by each forager bee and when the foragers return to the hive a waggle dance is performed by each forager. The other bees follow the foragers based on the fitness function of the best forager bee. This algorithm looks like the

Ant Colony Optimization (Dorigo and Stutzle 2004) algorithm but it does not use at all the concept of pheromone trails.

Contrary to the fact that there are many algorithms that are based on the foraging behaviour of the bees, the main algorithm proposed based on the marriage behaviour is the Honey Bees Mating Optimization (HBMO) algorithm, that was presented (Abbass 2001a, b). Since then, it has been used on a number of different applications (Afshar et al. 2007; Fathian et al. 2007; Haddad et al. 2006; Teo and Abbass 2003). The Honey Bees Mating Optimization algorithm simulates the mating process of the queen of the hive. The mating process of the queen begins when the queen flights away from the nest performing the mating flight during which the drones follow the queen and mate with her in the air (Abbass 2001a; Afshar et al. 2007). The algorithm is a swarm intelligence algorithm since it uses a swarm of bees where there are three kinds of bees, the queen, the drones and the workers. There is a number of procedures that can be applied inside the swarm. In the honey bees mating optimization algorithm, the procedure of mating of the queen with the drones is described. First, the queen is flying randomly in the air and, based on her speed and her energy, if she meets a drone then there is a possibility to mate with him. Even if the queen mates with the drone, she does not create directly a brood but stores the genotype (with the term "genotype" we mean some of the basic characteristics of the drones, i.e. part of the solution) of the drone in her spermatheca and the brood is created only when the mating flight has been completed. A crossover operator is used in order to create the broods. In a hive the role of the workers is simply the brood care (i.e. to feed them with the "royal jelly") and, thus, they are only a local search phase in the honey bees mating optimization algorithm. And thus, this algorithm combines both the mating process of the queen and one part of the foraging behavior of the honey bees inside the hive. If a brood is better (fittest) than the queen, then this brood replaces the queen.

In this paper, as there are not any competitive nature inspired methods based to Honey Bees Mating Optimization, at least to our knowledge, for the solution of the Vehicle Routing Problem (VRP) 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 (Abbass 2001a, b; Afshar et al. 2007; Fathian et al. 2007; Haddad et al. 2006) 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. More specifically, the proposed algorithm uses:

- The Multiple Phase Neighborhood Search–Greedy Randomized Adaptive Search Procedure (MPNS–GRASP) (Marinakis et al. 2007b) for the calculation of the initial population of bees and of the initial queen in order to have a more competitive queen.
- The Expanding Neighborhood Search (ENS) (Marinakis et al. 2005) as a local search strategy in order to have more effective and different between them workers. By using ENS, each brood has the possibility to select randomly the number of workers (local search phases) that will be used for the improvement of its solution.
- A new crossover operator based on an Adaptive Memory Procedure (Rochat and Taillard 1995) and on a uniform crossover operator in order to have fittest broods. This crossover operator combines the genotype of the queen and of more than one drones to produce a brood. The reason why such a crossover operator is used is because in real life the queen stores in her spermatheca, after the mating, the genotype of all drones and after returning to the hive she produces the broods. The adaptive memory procedure is used in order the queen to have the possibility to store from previous

selected good drones (in previous mating flights) part of their solutions in order to use them in a new mating flight and to produce more fittest broods.

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 Vehicle Routing Problem is presented. In Sect. 3 the proposed algorithm, the Honey Bees Mating Optimization (HBMOVRP), is presented and analyzed in detail. Computational results are presented and analyzed in Sect. 4 while in the last section conclusions and future research are given.

2 The Vehicle Routing Problem

The Vehicle Routing Problem (VRP) or the Capacitated 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. Let $G = (V,E)$ be a graph where $V = \{i_1, i_2,\ldots i_n\}$ is the vertex set (i_1 refers to the depot and the customers are indexed $i_2,\ldots,i_n$) and $E = \{(i_l,i_m):i_l,i_m \in V\}$ 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. A demand q_l and a service time st_l are associated with each customer node i_l. The demand q_1 and the service time st_1 which are referred to the demand and service time of the depot are set equal to zero. The travel cost and the travel time between customers i_l and i_m is c_{lm} and tt^k_{lm}, respectively, and T_k is the maximum time allowed for a route of vehicle k. 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, where the variable x^k_{lm} is equal to 1 if the arc (i_l, i_m) is traversed by vehicle k and 0 otherwise. The vehicle must start and finish its tour at the depot. In the following we present the mathematical formulation of the VRP (Bodin et al. 1983):

$$J = \min \sum_{l=1}^{n} \sum_{m=1}^{n} \sum_{k=1}^{K} c_{lm} x^k_{lm} \tag{1}$$

s.t.

$$\sum_{i_l=1}^{n} \sum_{k=1}^{K} x^k_{lm} = 1, \quad i_m = 2,\ldots,n \tag{2}$$

$$\sum_{i_m=1}^{n} \sum_{k=1}^{K} x^k_{lm} = 1, \quad i_l = 2,\ldots,n \tag{3}$$

$$\sum_{i_l=1}^{n} x^k_{lf} - \sum_{i_m=1}^{n} x^k_{fm} = 0, \quad k = 1,\ldots,K \quad i_f = 1,\ldots,n \tag{4}$$

$$\sum_{i_l=1}^{n} q_l \sum_{i_m=1}^{n} x_{lm}^k \leq Q_k, \quad k = 1, \ldots, K \tag{5}$$

$$\sum_{i_l=1}^{n} st_l^k \sum_{i_m=1}^{n} x_{lm}^k + \sum_{i_l=1}^{n} \sum_{i_m=1}^{n} tt_{lm}^k x_{lm}^k \leq T_k, \quad k = 1, \ldots, K \tag{6}$$

$$\sum_{i_m=2}^{n} x_{1m}^k \leq 1, \quad k = 1, \ldots, K \tag{7}$$

$$\sum_{i_l=2}^{n} x_{l1}^k \leq 1, \quad k = 1, \ldots, K \tag{8}$$

$$X \in S \tag{9}$$

$$x_{lm}^k = 0 \text{ or } 1, \quad \text{for all } i_l, i_m, k. \tag{10}$$

Objective function (1) states that the total distance is to be minimized. Equations 2 and 3 ensure that each demand node is served by exactly one vehicle. Route continuity is represented by (4), i.e. if a vehicle enters in a demand node, it must exit from that node. Equations 5 are the vehicle capacity constraints and 6 are the total elapsed route time constraints. Equations 7 and 8 guarantee that vehicle availability is not exceeded.

The Vehicle Routing Problem was first introduced by Dantzig and Ramser (1959). As it 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 were proposed. These techniques are classified into two main categories: Classical heuristics that were developed mostly between 1960 and 1990 (Bodin et al. 1983) and metaheuristics that were developed in the last fifteen years. In the 1990s a number of algorithms, known as metaheuristics were applied for the solution of the Vehicle Routing Problem. Simulated annealing (Osman 1993), tabu search (Gendreau et al. 1994; Osman 1993; Taillard 1993), together with a number of hybrid techniques are the main categories of the metaheuristic procedures. In the last 10 years a number of nature inspired methods have been applied for the solution of the Vehicle Routing Problem. The most common used nature inspired methods for the solution of this problem are genetic algorithms (Baker and Ayechew 2003; Berger and Barkaoui 2003; Marinakis et al. 2007a; Prins 2004), ant colony optimization (Reimann et al. 2002, 2004) and particle swarm optimization (Marinakis et al. 2007d). The reader can find more detailed descriptions of the algorithms used for the solution of VRP in the survey papers (Bodin et al. 1983; Fisher 1995; Gendreau et al. 1997, 2002; Laporte et al. 2000; Marinakis and Migdalas 2002; Tarantilis 2005) and in the books (Golden and Assad 1988; Toth and Vigo 2002).

3 Honey Bees Mating Optimization algorithm for the Vehicle Routing Problem

The proposed algorithm, the Honey Bees Mating Optimization Algorithm for the Vehicle Routing Problem (HBMOVRP), combines a number of different procedures. Each of them corresponds to a different phase of the mating process of the honey bees. Initially, we have to choose the population of the honey bees that will configure the initial hive. In the proposed algorithm, the initial population is created by using a modified version of the Greedy Randomized Adaptive Search Procedure (GRASP) (Feo and Resende 1995;

Resende and Ribeiro 2003), the Multiple Phase Neighborhood Search–GRASP (MPNS–GRASP) (Marinakis et al. 2005). 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 mating of the queen 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. The number of queens and the number of broods that will be born by all queens have to be defined. In this implementation of Honey Bees Mating Optimization (HBMO) algorithm, the number of queens is set equal to one, because 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 queen's spermatheca size. Then, we are ready to begin the mating flight of the queen. 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 cannot be equal to zero) and returns to her nest when the energy is less than a threshold value (*thres*) and spermatheca is not full (Abbass 2001a). A drone mates with a queen probabilistically using the following annealing function (Abbass 2001a, b):

$$Prob(D) = e^{\left[\frac{-\Delta(f)}{Speed(t)}\right]} \tag{11}$$

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:

$$Speed(t+1) = \alpha \times Speed(t) \tag{12}$$

$$energy(t+1) = \alpha \times energy(t) \tag{13}$$

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 Eq. 13 is different than the one proposed by Abbass (2001a, b) 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. 11. 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. A new crossover operator is developed 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 (Abbass 2001a; Afshar et al. 2007;

Fathian et al. 2007; Haddad et al. 2006; Teo and Abbass 2003) 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 heursitics (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 Table 1 while in the next paragraphs some procedures of the algorithm are explained in detail.

Table 1 Honey Bees Mating Optimization for the Vehicle Routing Problem

algorithm Honey Bees Mating Optimization for VRP
Initialization
Generate the initial population of the bees using MPNS–GRASP
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 (ENS strategy)
    if the brood's fitness is better than the queen's fitness
      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 Initial population

GRASP (Feo and Resende 1995; Marinakis et al. 2005; Resende and Ribeiro 2003) is an iterative method in which each iteration consists of two phases, a construction phase and a local search procedure. In the construction phase, a randomized greedy function is used to build up an initial solution. This randomized technique provides a feasible solution within each iteration and can be described as a process which stepwise adds one element at a time to a partial (incomplete) solution. The choice of the next element to be added is determined by ordering all elements in a candidate list, the Restricted Candidate List (RCL), with respect to a greedy function. The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidates in the list but not necessarily the top candidate. In the second phase, a local search is initialized from these points, and the final result is simply the best solution found over all searches.

The most important differences of MPNS–GRASP (Marinakis et al. 2007b) of the classic GRASP concerns the construction of the RCL list, the application of alternative greedy functions in each iteration instead of only one simple greedy function as in the classical approach. Moreover, in MPNS–GRASP a combination of greedy functions is also possible. The algorithm starts with one greedy function and if the results are not improving, an alternative greedy function is used instead. In these greedy functions, initially a Traveling Salesman Problem (TSP) is solved (Marinakis et al. 2005), disregarding the side constraints (capacity constraints and maximum route duration constraints) of the Vehicle Routing Problem. Subsequently, the solution of the TSP is converted to a solution of the VRP by adding the side constraints (Bodin et al. 1983). More precisely, the first vehicle route begins from the node that corresponds to the depot and moves to the next node (customer) based on the solution of the TSP, checking if the capacity of the vehicle or if the maximum route length of the vehicle are not violated. If any of these two constraints are violated, then the vehicle returns to the depot and a new route begins. The utilization of a simple local search in the second phase of the classical algorithm limits the chances of obtaining better solutions. Thus, MPNS–GRASP uses instead the Expanding Neighborhood Search, which is a very flexible local search strategy (Marinakis et al. 2005, 2007c). The steps of the MPNS–GRASP algorithm are given in Table 2.

3.2 Calculation of fitness function

In VRP, 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 high 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 high 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 cost and to subtract from this value the cost of each of the other individuals. By doing this the higher 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 cost 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.

Table 2 Multiple Phase Neighborhood Search–GRASP

Initialization

Select the set of the algorithms which will be used in the first main phase
of the MPNS–GRASP.
Select the set of local search algorithms which will be used in the second main phase
of the MPNS–GRASP.
Select the threshold value b_2
! b_2 = the acceptance quality of the solution using a specified tour construction method
in the first main phase of the algorithm after a prespecified number of iterations
Select the initial size of the RCL.

Main Algorithm

Set the number of iterations equal to zero.
Do while maximum number of iterations has not been reached:
Main Phase 1
Increase the iteration counter.
Construct the RCL.
Select randomly from the RCL the candidate element for inclusion to the partial
solution.
Call a greedy algorithm.
If the number of iterations is equal to the prespecified number **then**
If the quality of the best solution for the first main phase is larger
than the threshold b_2 **then**
Choose another construction heuristic
endif
endif
Main Phase 2
Call Expanding Neighborhood Search.
Enddo
Return the best solution

the symbol ! is used to denote the comments

3.3 Crossover operator

We propose a 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 (1995) as a part of a tabu search metaheuristic for the solution of the Vehicle Routing Problem. This procedure stores characteristics (tours in the Vehicle Routing Problem) of good solutions. Each time a new good solution is 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 a number possibilities:

1. The candidate for the adaptive memory solution is a previous best solution (queen) and its fitness function is at most 10% worst than the value of the current best solution.
2. The candidate for the adaptive memory solution is a member of the population (drone) and its fitness function is at most 10% worst than the value of the current best solution.
3. 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 brood is denoted by $b_i(t)$ (t is the iteration number), the solution of the queen is denoted by $q_i(t)$, the solution in the adaptive memory is denoted by $ad_i(t)$ and the solution of the drone by $d_i(t)$:

$$b_i(t) = \begin{cases} q_i(t), & \text{if } rand_i(0,1) \leq Cr_1 \\ ad_i(t), & \text{if } Cr_1 < rand_i(0,1) \leq Cr_2 \\ d_i(t), & \text{otherwise.} \end{cases} \tag{14}$$

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

3.4 Workers—local search heuristics

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 (Marinakis et al. 2005). Expanding Neighborhood Search (ENS) is a metaheuristic algorithm (Marinakis et al. 2005) 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.

3.4.1 Circle Restricted Local Search Moves

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 into circles around the candidate for deletion edges.

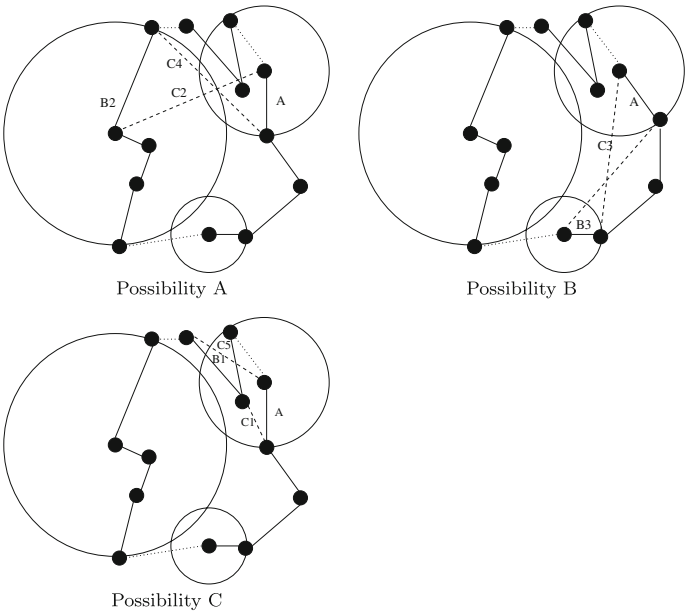

Fig. 1 Circle Restricted Local Search Moves Strategy

In the following, a description of the Circle Restricted Local Search Moves Strategy for a 2-opt trial move (Lin 1965) is presented. In this case, there are three possibilities based on the costs of the candidates for deletion and inclusion edges:

- If both new edges increase in cost, a 2-opt trial move can not reduce the cost of the tour [e.g., in Fig. 1 (Possibility A), for both new edges the costs $C2$ and $C4$ are greater than the costs $B2$ and A of both old edges].
- If one of the two new edges has cost greater than the sum of the costs of the two old edges, a 2-opt trial move, again, can not reduce the cost of the tour [e.g. in Fig. 1 (Possibility B), the cost of the new edge $C3$ is greater than the sum of the costs $A + B3$ of the old edges].
- The only case for which a 2-opt trial move can reduce the cost of the tour is when at least one new edge has cost less than the cost of one of the two old edges [e.g., in Fig. 1 (Possibility C), the cost $C1$ of the new edge is less than the cost of the old edge A] and the other edge has cost less than the sum of the costs of the two old edges [e.g., $C5 < A + B1$ in Fig. 1 (Possibility C)].

Taking these observations into account, the Circle Restricted Local Search Moves strategy restricts the search to edges where one of their end-nodes is inside a circle with radius length at most equal to the sum of the costs (lengths) of the two candidates for deletion edges.

3.4.2 Different local search strategies in ENS

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 (1972) and recently by Hansen and Mladenovic (2001). 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.

On the other hand, the Expanding Neighborhood Search Method starts with one pre-specified length of the radius of the circle of the CRLSM strategy. Inside this circle, a number of different local search strategies are applied until all the possible trial moves have been explored and the solution cannot further be improved in this neighborhood. Subsequently, the length of the radius of the circle is increased and, again, the same procedure is repeated until the stopping criterion is activated. The main differences of ENS from the other two methods is the use of the Circle Restricted Local Search Moves Strategy which restricts the search in circles around the candidates for deletion edges and the more sophisticated way that the local search strategy can be changed inside the circles.

The local search strategies for the Vehicle Routing Problem are distinguished between local search strategies for a single route and local search strategies for multiple routes. The local search strategies that are chosen and belong to the category of the single route interchange (strategies that try to improve the routing decisions) are the well known methods for the TSP, the 2-opt and the 3-opt (Bodin et al. 1983). In the single route interchange all the routes have been created in the initial phase of the algorithm. The Local Search Strategies for Multiple Route Interchange try to improve the assignment decisions. This, of course, increases the complexity of the algorithms but gives the possibility to improve even more the solution. The multiple route interchange local search strategies that are used are the 1-0 relocate, 2-0 relocate, 1-1 exchange, 2-2 exchange and crossing (Toth and Vigo 2002).

3.4.3 Expanding strategy in ENS

In Expanding Neighborhood Search strategy, the size of the neighborhood is expanded in each external iteration (Marinakis et al. 2005). Each different length of the neighborhood constitutes an external iteration. Initially, the size of the neighborhood, f, is defined based on the Circle Restricted Local Search Moves strategy, for example $f = A/2$, where A is the cost of one of the candidates for deletion edges. For the selected size of the neighborhood, a number of different local search strategies are applied until all the possible trial moves have been explored and the solution can not further be improved in this neighborhood.

The local search strategies are changed based on two conditions, i.e. if the current local search strategy finds a local optimum or if the quality of the current solution remains greater than the threshold number b_1 for a number of internal iterations. (The quality is given in terms of the relative deviation from the best known solution, that is $\omega_{HBMOVRP} = \frac{(c_{HBMOVRP} - c_{BKS})}{c_{BKS}}\%$, where $c_{HBMOVRP}$ denotes the cost of the solution found by HBMOVRP and c_{BKS} is the cost of the best known solution.) If the quality of the solution is less than a threshold number e the algorithm stops, otherwise the neighborhood is expanded by increasing the length of the radius of the CRLSM strategy f by a percentage θ (θ is determined empirically after thorough investigation and is set equal to 10%) and the algorithm continues. When the length of the radius of the CRLSM strategy is equal to A, the length continues to increase until the length becomes equal to $A + B$, where B is the length of the other candidate for deletion edge. If the length of the radius of the CRLSM

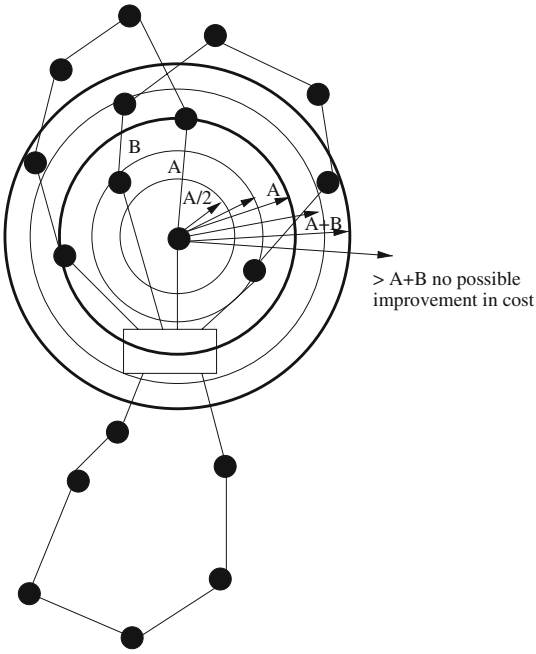

Fig. 2 Expanding Neighborhood Search Strategy

Table 3 Expanding Neighborhood Search

```
f = A/2, m = 0 ! m = index for the local search methods
do while (ω > e or f < A + B)    !ω = the quality of the solution
   m = 1
   Call first local search strategy
   if ω < e then
      STOP with the current solution
   else
      do while (m < M1)
      ! M1 = the number of local search strategy
         if ω > b1 or a local optimum is found then
            Change local search method
            m = m + 1
            if ω < e then
               STOP with the current solution
            endif
         endif
      enddo
   endif
   f = 1.1*f
   Update b1 and e
enddo
```

the symbol ! is used to denote the comments

strategy is equal to $A + B$, and the algorithm has reached the maximum number of iterations, then the algorithm terminates with the current solution. In Fig. 2, the Expanding Neighborhood Search Method is presented. In Table 3 the pseudocode of ENS is given.

4 Computational results

The whole algorithmic approach was implemented in Fortran 90 and was compiled using the Lahey f95 compiler on a Centrino Mobile Intel Pentium M 750 at 1.86 GHz, running Suse Linux 9.1. 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 1,000, size of queen's spermatheca equal to 50, number of broods equal to 50, α equal to 0.9, number of workers (w) equal to 20, ($w_1 = 7$, $w_2 = 13$), size of RCL equal to 50.

The algorithm was tested on two sets of benchmark problems. The 14 benchmark problems proposed by Christofides et al. (1979) and the 20 large scale vehicle routing problems proposed by Golden et al. (1998). Each instance of the first set contains between 51 and 200 nodes including the depot. Each problem includes capacity constraints while the problems 6-10, 13 and 14 have, also, maximum route length restrictions and non zero service times. For the first 10 problems, nodes are randomly located over a square, while for the remaining ones, nodes are distributed in clusters and the depot is not centered. The second set of instances contains between 200 and 483 nodes including the depot. Each problem instance includes capacity constraints while the first eight have, also, maximum route length restrictions but with zero service times. The efficiency of the HBMOVRP 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, $\omega_{HBMOVRP}$.

In the first column of Tables 4 and 5 the number of nodes of each instance is presented, while in the second, third and fourth columns the most important characteristics of the instances, namely the maximum capacity of the vehicles (column 2), the maximum tour length (m.t.l.—column 3) of each vehicle and the service time (s.t.—column 4) of each customer, are presented. In the last four columns, the results of the proposed algorithm (column 5), the best known solution (BKS—column 6), the quality of the solution of the proposed algorithm ($\omega_{HBMOVRP}$—column 7) and the CPU time of each instance of the proposed algorithm (column 8) are presented, respectively. It can be seen from Table 4, that the algorithm, in 10 of the 14 instances of the first set has reached the best known solution. For the other four instances the quality of the solutions is between 0.07% and 0.16% and the average quality for the fourteen instances is 0.029%. For the 20 large scale vehicle routing problems (Table 5) the algorithm has found the best known solution in two of them, for the rest the quality is between 0.13% and 0.65% and the average quality of the solutions is 0.40%. Also, in these tables the computational time needed (in minutes) for finding the best solution by HBMOVRP is presented. The CPU time needed is significantly low for the first set of instances and only for two instances (instance 5 and 10) is somehow increased but still is very efficient. In the second set of instances, the problems are more complicated and, thus, the computational time is increased but is still less than 10 min in all instances. These results denote the efficiency of the proposed algorithm. It should be noted that we would like to present a very fast and effective algorithm and thus the choice of the parameters (like the number of drones and the number of mating flights) was performed in such a way that the algorithm to combine a fast convergence with as good as possible results. This issue sometimes led the algorithm not to find the optimum. If we increase the number of drones and the number of mating flights the algorithm improves even more the solutions but then the algorithm finds these solutions in more computational time.

Table 4 Results of HBMOVRP in Christofides benchmark instances

Nodes	Cap.[a]	m.t.l.[b]	s.t.[c]	HBMOVRP	BKS[d]	$\omega_{HBMOVRP}$ (%)	CPU (min)
51	160	∞	0	524.61	524.61	0.00	0.04
76	140	∞	0	835.26	835.26	0.00	0.23
101	200	∞	0	826.14	826.14	0.00	0.29
151	200	∞	0	1028.42	1028.42	0.00	1.08
200	200	∞	0	1292.57	1291.45	0.10	2.07
51	160	200	10	555.43	555.43	0.00	0.06
76	140	160	10	909.68	909.68	0.00	0.35
101	200	230	10	867.31	865.94	0.16	0.95
151	200	200	10	1163.52	1162.55	0.08	1.47
200	200	200	10	1395.85	1395.85	0.00	2.98
121	200	∞	0	1042.11	1042.11	0.00	0.52
101	200	∞	0	819.56	819.56	0.00	0.37
121	200	720	50	1542.21	1541.14	0.07	0.40
101	200	1040	90	866.37	866.37	0.00	0.36

[a] Maximum capacity of the vehicles

[b] Maximum route length of each vehicle

[c] Service time of each customer

[d] Best known solution

In order to give the significance and to prove the contribution of each of the characteristics (metaheuristics used) in the HBMOVRP, we implemented each of the main phases separately and we compared their results with the results of HBMOVRP. There are two non evolutionary algorithms, the Expanding Neighborhood Search (ENS) (columns 1 and 2 in Tables 6 and 7) and the Multiple Phase Neighborhood Search–GRASP (MPNS–GRASP) (columns 3 and 4 in Tables 6 and 7) and one evolutionary algorithm, the Hybrid Genetic GRASP-ENS (HybGEN) (columns 5 and 6 in Tables 6 and 7). Expanding Neighborhood Search (ENS) is used as it was described in Sect. 3.4. The MPNS–GRASP is used as described in Sect. 3.1 without using any of the characteristics of the Honey Bees Mating Optimization algorithm. The HybGEN algorithm is a Hybrid Genetic algorithm where the ENS is used in the mutation phase of a simple genetic algorithm in order to improve the solutions. The reasons why HybGEN algorithm is used in the comparisons is because we would like to show the fact that HBMOVRP algorithm gives superior results compared to another population based algorithm and because HybGEN algorithm uses the same crossover operator that we use in HBMOVRP. In all implementations, the parameters were chosen in such a way that in all algorithms to have the same number of function evaluations. The parameters that do not affect the number of the function evaluations were set equal to the parameters of HBMOVRP and the local search strategies were the same as in HBMOVRP.

In Tables 6 and 7, the cost and the quality of the solutions given by the algorithms are presented. As it can be observed from Tables 6 and 7 the results are significantly improved with the use of the proposed algorithm. More precisely, the improvement in the quality of the results of the proposed method from the ENS algorithm is between 0.00% and 1.82% in the Christofides benchmark instances with average improvement equal to 0.541% and is between 0.386% and 3.48% in the Golden benchmark instances with average improvement

Table 5 Results of HBMOVRP in the 20 benchmark Golden instances

Nodes	Cap.[a]	m.t.l.[b]	s.t.[c]	HBMOVRP	BKS[d]	$\omega_{HBMOVRP}$ (%)	CPU (min)
240	550	650	0	5645.51	5627.54	0.32	1.78
320	700	900	0	8458.72	8447.92	0.13	2.17
400	900	1,200	0	11086.21	11036.22	0.45	6.42
480	1,000	1,600	0	13675.23	13624.52	0.37	7.21
200	900	1,800	0	6460.98	6460.98	0.00	1.23
280	900	1,500	0	8467.57	8412.8	0.65	1.51
360	900	1,300	0	10212.23	10195.56	0.16	2.34
440	900	1,200	0	11710.53	11663.55	0.40	6.57
255	1,000	∞	0	586.52	583.39	0.54	1.37
323	1,000	∞	0	745.21	742.03	0.43	2.83
399	1,000	∞	0	924.34	918.45	0.64	3.21
483	1,000	∞	0	1112.52	1107.19	0.48	8.17
252	1,000	∞	0	863.79	859.11	0.54	3.25
320	1,000	∞	0	1088.27	1081.31	0.64	2.21
396	1,000	∞	0	1352.57	1345.23	0.55	7.59
480	1,000	∞	0	1629.28	1622.69	0.41	9.80
240	200	∞	0	711.09	707.79	0.47	2.43
300	200	∞	0	1003.28	998.73	0.46	2.57
360	200	∞	0	1366.86	1366.86	0.00	3.29
420	200	∞	0	1826.54	1821.15	0.30	5.23

[a] Maximum capacity of the vehicles

[b] Maximum route length of each vehicle

[c] Service time of each customer

[d] Best known solution

equal to 1.53%. The improvement in the quality of the results of the proposed method from the MPNS–GRASP algorithm is between 0.00% and 1.68% in the Christofides benchmark instances with average improvement equal to 0.365% and is between 0.219% and 2.22% in the Golden benchmark instances with average improvement equal to 0.91%. This issue is very important because it is proved that the procedure of the Honey Bees Mating Optimization improves significantly the results of the MPNS–GRASP algorithm. The improvement in the quality of the results of the proposed method from the HybGEN algorithm is between 0.00% and 0.51% in the Christofides benchmark instances with average improvement equal to 0.092% and is between 0.00% and 1.87% in the Golden benchmark instances with average improvement equal to 0.485%. Concerning the HybGEN and the HBMOVRP it should be noted that although they both are population based algorithms and they both use the same crossover operator and ENS, the HBMOVRP gives significantly better results than the HybGEN. This last issue is very important because it is proved that each phase of the algorithm is needed in order to have as good results as possible. Also, it should be noted that the use of the HBMOVRP algorithm without MPNS–GRASP and ENS led to an important increase of the time that the algorithm needed to perform the same function evaluations with the HBMOVRP. Thus, each phase of the proposed algorithm is very important in the overall performance of the algorithm. Thus, the

Table 6 Comparison of the proposed algorithm with ENS, MPNS–GRASP and HybGEN in the 14 Christofides benchmark instances

ENS		MPNS–GRASP		HybGEN		HBMOVRP	
Cost	ω (%)	Cost	ω (%)	Cost	ω (%)	Cost	ω (%)
524.61	0.00	524.61	0.00	524.61	0.00	524.61	0.00
837.56	0.27	836.39	0.13	835.26	0.00	835.26	0.00
826.14	0.00	826.14	0.00	826.14	0.00	826.14	0.00
1034.48	0.58	1032.24	0.37	1028.42	0.00	1028.42	0.00
1316.18	1.91	1314.25	1.78	1299.21	0.61	1292.57	0.10
555.43	0.00	555.43	0.00	555.43	0.00	555.43	0.00
909.68	0.00	909.68	0.00	909.68	0.00	909.68	0.00
868.27	0.26	867.31	0.16	867.31	0.16	867.31	0.16
1178.86	1.40	1175.86	1.14	1165.13	0.22	1163.52	0.08
1416.14	1.45	1412.11	1.16	1402.27	0.46	1395.85	0.00
1043.53	0.13	1042.11	0.00	1042.11	0.00	1042.11	0.00
824.57	0.61	821.12	0.19	819.56	0.00	819.56	0.00
1551.24	0.65	1548.53	0.47	1545.02	0.25	1542.21	0.07
872.14	0.66	868.62	0.25	866.37	0.00	866.37	0.00

Table 7 Comparison of the proposed algorithm with ENS, MPNS–GRASP and HybGEN in the 20 Golden instances

ENS		MPNS–GRASP		HybGEN		HBMOVRP	
Cost	ω (%)	Cost	ω (%)	Cost	ω (%)	Cost	ω (%)
5740.45	2.01	5715.19	1.56	5689.58	1.10	5645.51	0.32
8518.21	0.83	8490.15	0.50	8459.73	0.14	8458.72	0.13
11185.24	1.35	11144.39	0.98	11101.12	0.59	11086.21	0.45
13785.28	1.18	13752.24	0.94	13698.17	0.54	13675.23	0.37
6485.98	0.39	6475.19	0.22	6460.98	0.00	6460.98	0.00
8503.35	1.08	8492.28	0.94	8470.64	0.69	8467.57	0.65
10296.18	0.99	10275.17	0.78	10215.14	0.19	10212.23	0.16
12021.18	3.07	11918.15	2.18	11878.21	1.84	11710.53	0.40
595.27	2.04	589.94	1.12	586.87	0.60	586.52	0.54
759.38	2.34	749.15	0.96	746.56	0.61	745.21	0.43
937.18	2.04	935.23	1.83	925.52	0.77	924.34	0.64
1151.13	3.97	1137.17	2.71	1133.28	2.36	1112.52	0.48
882.17	2.68	875.14	1.87	868.17	1.05	863.79	0.54
1101.12	1.83	1098.95	1.18	1094.87	1.25	1088.27	0.64
1382.34	2.76	1369.16	1.78	1364.28	1.42	1352.57	0.55
1658.21	2.19	1651.14	1.75	1644.17	1.32	1629.28	0.41
719.26	1.62	715.16	1.04	712.18	0.62	711.09	0.47
1025.18	2.65	1013.17	1.45	1008.19	0.95	1003.28	0.46
1395.16	2.07	1384.18	1.27	1378.21	0.83	1366.86	0.00
1848.25	1.49	1835.18	0.77	1835.17	0.77	1826.54	0.30

use of the Honey Bees Mating Optimization algorithm in a hybridization algorithm gives very good results.

In order to give the significance and to prove the contribution of the new crossover operator and of the use of different workers in each brood in the HBMOVRP, we implemented three different versions of the honey bees mating optimization algorithm where in the first one neither the new crossover operator nor the ENS strategy in the workers are used, in the second only the new crossover operator is used and in the third version only the ENS strategy in the workers is used. The results of these three implementations are compared with the results of the proposed algorithm in Tables 8 and 9. In all implementations we use for the initial solutions the Multiple Phase Neighborhood Search–GRASP. The results of the first implementation (HBMO1) are presented in columns 1 and 2 in Tables 8 and 9, while the results of the second implementation (HBMO2) are presented in columns 3 and 4 in Tables 8 and 9, and the results of the third implementation (HBMO3) are presented in columns 5 and 6 in Tables 8 and 9. In all implementations, the parameters were chosen in such a way that in all algorithms the same number of function evaluations to be made. The parameters that do not affect the number of the function evaluations were set equal to the parameters of HBMOVRP and the local search strategies were the same as in HBMOVRP.

In Tables 8 and 9, the cost and the quality of the solutions given by the algorithms are presented. As it can be observed from Tables 8 and 9 the results are significantly improved with the use of the proposed algorithm. More precisely, the improvement in the quality of the results of the proposed method from the HBMO1 algorithm is between 0.00% and 1.52% in the Christofides benchmark instances with average improvement equal to 0.33% and is between 0.16% and 2.17% in the Golden benchmark instances with average improvement equal to 0.84%. The improvement in the quality of the results of the proposed method from the HBMO2 algorithm is between 0.00% and 0.49% in the Christofides benchmark instances with average improvement equal to 0.11% and is between 0% and

Table 8 Comparison of the proposed algorithm with other implementations of Honey Bees Mating Optimization algorithm in the 14 Christofides benchmark instances

HBMO1		HBMO2		HBMO3		HBMOVRP	
Cost	ω (%)	Cost	ω (%)	Cost	ω (%)	Cost	ω (%)
524.61	0.00	524.61	0.00	524.61	0.00	524.61	0.00
836.21	0.11	835.26	0.00	835.26	0.00	835.26	0.00
826.14	0.00	826.14	0.00	826.14	0.00	826.14	0.00
1031.18	0.27	1030.21	0.17	1029.28	0.08	1028.42	0.00
1312.21	1.62	1298.27	0.54	1301.18	0.77	1292.57	0.10
555.43	0.00	555.43	0.00	555.43	0.00	555.43	0.00
909.68	0.00	909.68	0.00	909.68	0.00	909.68	0.00
867.31	0.16	867.31	0.16	867.31	0.16	867.31	0.16
1172.12	0.82	1169.18	0.57	1167.21	0.40	1163.52	0.08
1411.98	1.16	1399.21	0.24	1397.28	0.10	1395.85	0.00
1042.11	0.00	1042.11	0.00	1042.11	0.00	1042.11	0.00
820.98	0.17	819.71	0.02	820.76	0.15	819.56	0.00
1547.21	0.39	1544.21	0.20	1543.37	0.14	1542.21	0.07
868.59	0.26	866.37	0.00	866.51	0.02	866.37	0.00

Table 9 Comparison of the proposed algorithm with other implementations of Honey Bees Mating Optimization algorithm in the 20 Golden instances

HBMO1		HBMO2		HBMO3		HBMOVRP	
cost	ω (%)	cost	ω (%)	cost	ω (%)	cost	ω (%)
5708.21	1.43	5688.17	1.08	5675.71	0.86	5645.51	0.32
8488.21	0.52	8472.51	0.33	8469.37	0.29	8458.72	0.13
11136.59	0.91	11097.39	0.55	11087.18	0.46	11086.21	0.45
13749.49	0.92	13688.29	0.47	13699.51	0.55	13675.23	0.37
6471.29	0.16	6460.98	0.00	6461.21	0.00	6460.98	0.00
8491.20	0.93	8471.17	0.69	8474.37	0.73	8467.57	0.65
10271.29	0.88	10267.35	0.84	10254.71	0.72	10212.23	0.16
11901.57	2.21	11809.57	1.42	11798.31	1.33	11710.53	0.40
588.31	0.84	587.01	0.62	586.95	0.61	586.52	0.54
748.32	0.91	745.38	0.52	745.21	0.49	745.21	0.43
934.17	1.71	927.12	0.94	927.08	0.94	924.34	0.64
1136.54	2.65	1120.28	1.18	1119.12	1.08	1112.52	0.48
875.05	1.86	863.81	0.55	863.72	0.54	863.79	0.54
1098.08	1.55	1089.07	0.72	1088.52	0.67	1088.27	0.64
1368.72	1.75	1354.17	0.66	1358.21	0.96	1352.57	0.55
1650.91	1.74	1631.01	0.51	1633.14	0.64	1629.28	0.41
715.09	1.03	711.21	0.48	711.09	0.47	711.09	0.47
1013.17	1.57	1005.29	0.78	1006.12	0.86	1003.28	0.46
1379.45	0.92	1366.98	0.01	1366.86	0.00	1366.86	0.00
1835.01	0.82	1829.10	0.50	1827.34	0.40	1826.54	0.30

0.85% in the Golden benchmark instances with average improvement equal to 0.22%. The improvement in the quality of the results of the proposed method from the HBMO3 algorithm is between 0.00% and 0.66% in the Christofides benchmark instances with average improvement equal to 0.10% and is between 0.00% and 0.75% in the Golden benchmark instances with average improvement equal to 0.20%. As it can be observed from the previous results the use of both ENS and the new crossover operator improves significantly the results obtained from the proposed algorithm compared to the results of the classic honey bees mating optimization algorithm. Also, if only one of the two procedures is included in the honey bees mating optimization algorithm the results are slightly inferior to the results of the proposed algorithm but are almost equivalent between them. This last issue is very important because it proves that the usage of both proposed procedures is significant for the effectiveness of the algorithm.

The results obtained by the proposed algorithm are also compared to the results of the 10 most efficient metaheuristic algorithms and the 10 most efficient nature inspired algorithms that have ever been presented for the Vehicle Routing Problem. In Tables 10 and 11, the ranking of all algorithms is presented (in Golden instances we didn't find 10 nature inspired algorithms and thus we present the six algorithms used for the solution of these instances). The proposed algorithm is ranked in the second place among the 10 most efficient metaheuristics and in the first place among the nature inspired methods used for the solution of the VRP in Christofides instances. In the set of the large scale vehicle routing instances the algorithm is ranked in the second place both in metaheuristics and

Table 10 Comparison of other metaheuristics and nature inspired algorithms with HBMOVRP in Christofides benchmark instances

Rank	Algorithm	Quality (%)	n_{opt}	CPU (min)	Computer used
Metaheuristic algorithms					
1	RT (Rochat and Taillard 1995)	0.00	14	Not mentioned	Silicon Graphics 100MHz
2	HBMOVRP	0.029	10	0.79	Pentium M 750 at 1.86 GHz
3	AGES best (Mester and Braysy 2007)	0.03	13	0.27	Pentium IV 2GHz
4	Taillard (1993)	0.051	12	Not mentioned	Silicon Graphics 100MHz
5	HybPSO (Marinakis et al. 2007d)	0.084	7	0.80	Pentium M 750 at 1.86 GHz
6	Best-Prins (Prins 2004)	0.085	10	5.2	Pentium 1000 MHz
7	HybGEN	0.122	10	0.95	Pentium III 667MHz
8	Best-SEPAS (Tarantilis 2005)	0.182	11	6.6	Pentium II 400 MHz
9	St-SEPAS (Tarantilis 2005)	0.195	9	5.6	Pentium II 400 MHz
10	Best-TABUROUTE (Gendreau et al. 1994)	0.198	8	46.8	Silicon Graphics 36 MHz
Nature inspired methods					
1	HBMOVRP	0.029	10	0.79	Pentium M 750 at 1.86 GHz
2	AGES best (Mester and Braysy 2007)	0.03	13	0.27	Pentium IV 2GHz
3	HybPSO (Marinakis et al. 2007d)	0.084	7	0.80	Pentium M 750 at 1.86 GHz
4	Best-Prins (Prins 2004)	0.085	10	5.2	Pentium 1000 MHz
5	HybGEN	0.122	10	0.95	Pentium III 667MHz
6	Stand-Prins (Prins 2004)	0.235	8	5.2	Pentium 1000 MHz
7	RSD (Reimann et al. 2002)	0.383	6	7.7	Pentium 900 MHz
8	VRPBilevel (Marinakis et al. 2007a)	0.479	7	0.76	Pentium III 667MHz
9	D-ants (Reimann et al. 2004)	0.481	5	3.28	Pentium 900 MHz
10	Stand-HGA (Berger and Barkaoui 2003)	0.485	6	21.3	Pentium 400 MHz

nature inspired methods. In these tables, the number of optimally solved instances for every method is, also, presented (n_{opt}). Finally, in the last two columns of these tables, the average CPU time (in minutes) and the computers used for the metaheuristic and nature inspired algorithms of the comparisons are presented. It should be noted that a fair comparison in terms of computational efficiency is difficult because the computational speed is affected, mainly, by the compiler and the hardware that are used.

5 Conclusions

In this paper, a nature inspired approach was introduced for the effective handling of the Vehicle Routing Problem (VRP). More specifically, a hybrid algorithmic nature inspired methodology was proposed, namely the Honey Bees Mating Optimization algorithm for the VRP (HBMOVRP) that gave remarkable results both to quality and computational efficiency. The algorithm was applied in two sets of benchmark instances and gave very

Table 11 Comparison of other metaheuristics and nature inspired algorithms with HBMOVRP in Golden benchmark instances

Rank	Algorithm	Quality (%)	n_{opt}	CPU (min)	Computer used
Metaheuristic algorithms					
1	AGES best (Mester and Braysy 2007)	0.00	20	0.63	Pentium IV 2GHz
2	HBMOVRP	0.40	2	4.06	Pentium M 750 at 1.86 GHz
3	D-ants (Reimann et al. 2004)	0.59	4	49.33	Pentium 900 MHz
4	Stand-SEPAS (Tarantilis 2005)	0.60	2	45.48	Pentium II 400 MHz
5	BoneRoute (Tarantilis and Kiranoudis 2002)	0.74	1	42.05	Pentium 400 MHz
6	HybGEN	0.91	1	3.38	Pentium III 667MHz
7	Best-Prins (Prins 2004)	0.91	6	66.6	Pentium 1000 MHz
8	VRTR (Li et al. 2005)	1.05	0	0.97	Athlon 1 GHz
9	VRPBilevel (Marinakis et al. 2007a)	1.22	0	3.44	Pentium III 667MHz
10	LBTA (Tarantilis et al. 2002)	1.94	1	6.8	Pentium 233 MHz
Nature inspired methods					
1	AGES best (Mester and Braysy 2007)	0.00	20	0.63	Pentium IV 2GHz
2	HBMOVRP	0.40	2	4.06	Pentium M 750 at 1.86 GHz
3	D-ants (Reimann et al. 2004)	0.59	4	49.33	Pentium 900 MHz
4	HybGEN	0.91	1	3.38	Pentium III 667MHz
5	Prins (2004)	0.91	6	66.6	Pentium 1000 MHz
6	VRPBilevel (Marinakis et al. 2007a)	1.22	0	3.44	Pentium III 667MHz

satisfactory results. More specifically, in the set with the classic benchmark instances proposed by Christofides, the average quality is 0.029% and, thus, the algorithm is ranked in the second place among the most known metaheuristic algorithms and in the first place among the nature inspired methods used for the solution of the VRP in the literature. In the second set of instances proposed by Golden the average quality is 0.40% and, thus, the algorithm is ranked in the second place among the most known metaheuristic algorithms and in the second place among the nature inspired methods used for the solution of the VRP in the literature.

References

Abbass HA (2001a) 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

Abbass HA (2001b) Marriage in honey-bee optimization (MBO): a haplometrosis polygynous swarming approach. In: The congress on evolutionary computation (CEC2001), Seoul, Korea, May 2001, pp 207–214

Afshar A, Bozog Haddad O, Marino MA, Adams BJ (2007) Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation. J Frankl Inst 344:452–462

Baker BM, Ayechew MA (2003) A genetic algorithm for the vehicle routing problem. Comput Oper Res 30(5):787–800

Baykasoglu A, Ozbakor L, Tapkan P (2007) Artificial bee colony algorithm and its application to generalized assignment problem. In: Chan FTS, Tiwari MK (eds) Swarm intelligence, focus on ant and particle swarm optimization. I-Tech Education and Publishing, Vienna, pp 113–144

Berger J, Barkaoui M (2003) A hybrid genetic algorithm for the capacitated vehicle routing problem. In: Proceedings of the genetic and evolutionary computation conference, Chicago, pp 646–656

Bodin L, Golden B, Assad A, Ball M (1983) The state of the art in the routing and scheduling of vehicles and crews. Comput Oper Res 10:63–212
Christofides N, Mingozzi A, Toth P (1979) The vehicle routing problem. In: Christofides N, Mingozzi A, Toth P, Sandi C (eds) Combinatorial optimization. Wiley, Chichester
Dantzig GB, Ramser JH (1959) The truck dispatching problem. Manag Sci 6(1):80–91
Dorigo M, Stutzle T (2004) Ant colony optimization. A Bradford book. The MIT Press, Cambridge
Drias H, Sadeg S, Yahi S (2005) Cooperative bees swarm for solving the maximum weighted satisfiability problem. In: IWAAN international work conference on artificial and natural neural networks, LNCS, vol 3512, pp 318–325
Fathian M, Amiri B, Maroosi A (2007) Application of honey bee mating optimization algorithm on clustering. Appl Math Comput. doi:10.1016/j.amc.2007.02.029
Feo TA, Resende MGC (1995) Greedy randomized adaptive search procedure. J Glob Optim 6:109–133
Fisher ML (1995) Vehicle routing. In: Ball MO, Magnanti TL, Momma CL, Nemhauser GL (eds) Network routing, handbooks in operations research and management science, vol 8. North Holland, Amsterdam, pp 1–33
Garfinkel R, Nemhauser G (1972) Integer programming. Wiley, New York
Gendreau M, Hertz A, Laporte G (1994) A tabu search heuristic for the vehicle routing problem. Manag Sci 40:1276–1290
Gendreau M, Laporte G, Potvin J-Y (1997) Vehicle routing: modern heuristics. In: Aarts EHL, Lenstra JK (eds) Local search in combinatorial optimization. Wiley, Chichester, pp 311–336
Gendreau M, Laporte G, Potvin J-Y (2002) Metaheuristics for the capacitated VRP. In: Toth P, Vigo D (eds) The vehicle routing problem, monographs on discrete mathematics and applications. Siam, Philadelphia, pp 129–154
Golden BL, Assad AA (1988) Vehicle routing: methods and studies. North Holland, Amsterdam
Golden BL, Wasil EA, Kelly JP, Chao IM (1998) The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In: Crainic TG, Laporte G (eds) Fleet management and logistics. Kluwer, Boston, pp 33–56
Haddad OB, Afshar A, Marino MA (2006) Honey-bees mating optimization (HBMO) algorithm: a new heuristic approach for water resources optimization. Water Resour Manag 20:661–680
Hansen P, Mladenovic N (2001) Variable neighborhood search: principles and applications. Eur J Oper Res 130:449-467
Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim. doi:10.1007/s10898-007-9149-x
Karaboga D, Basturk B (2008) On the performance of artificial bee colony (ABC) algorithm. Appl Soft Comput 8:687–697
Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of 1995 IEEE international conference on neural networks, vol 4, pp 1942–1948
Laporte G, Gendreau M, Potvin J-Y, Semet F (2000) Classical and modern heuristics for the vehicle routing problem. Int Trans Oper Res 7:285–300
Li F, Golden B, Wasil E (2005) Very large-scale vehicle routing: new test problems, algorithms and results. Comput Oper Res 32(5):1165–1179
Lin S (1965) Computer solutions of the traveling salesman problem. Bell Syst Tech J 44:2245–2269
Marinakis Y, Migdalas A (2002) Heuristic solutions of vehicle routing problems in supply chain management. In: Pardalos PM, Migdalas A, Burkard R (eds) Combinatorial and global optimization. World Scientific, Singapore, pp 205–236
Marinakis Y, Migdalas A, Pardalos PM (2005) Expanding neighborhood GRASP for the traveling salesman problem. Comput Optim Appl 32:231–257
Marinakis Y, Migdalas A, Pardalos PM (2007a) A new bilevel formulation for the vehicle routing problem and a solution method using a genetic algorithm. J Glob Optim 38:555–580
Marinakis Y, Migdalas A, Pardalos PM (2007b) Multiple phase neighborhood search GRASP based on Lagrangean relaxation and random backtracking Lin Kernighan for the traveling salesman problem. J Comb Optim. doi:10.1007/s10878-007-9104-2
Marinakis Y, Migdalas A, Pardalos PM (2007c) Expanding neighborhood search—GRASP for the probabilistic traveling salesman problem. Optim Lett. doi:10.1007/s11590-007-0064-3
Marinakis Y, Marinaki M, Dounias G (2007d) A hybrid particle swarm optimization algorithm for the vehicle routing problem (submitted)
Mester D, Braysy O (2007) Active-guided evolution strategies for large-scale capacitated vehicle routing problems. Comput Oper Res 34(10):2964–2975
Osman IH (1993) Metastrategy simulated annealing and tabu search algorithms for combinatorial optimization problems. Ann Oper Res 41:421–451

Pham DT, Kog E, Ghanbarzadeh A, Otri S, Rahim S, Zaidi M (2006) The bees algorithm—a novel tool for complex optimization problems. In: IPROMS 2006 proceeding 2nd international virtual conference on intelligent production machines and systems. Elsevier, Oxford

Prins C (2004) A simple and effective evolutionary algorithm for the vehicle routing problem. Comput Oper Res 31:1985–2002

Reimann M, Stummer M, Doerner K (2002) A savings based ant system for the vehicle routing problem. In: Proceedings of the genetic and evolutionary computation conference, New York, pp 1317–1326

Reimann M, Doerner K, Hartl RF (2004) D-ants: savings based ants divide and conquer the vehicle routing problem. Comput Oper Res 31(4):563–591

Resende MGC, Ribeiro CC (2003) Greedy randomized adaptive search procedures. In: Glover F, Kochenberger GA (eds) Handbook of metaheuristics. Kluwer, Boston, pp 219–249

Rochat Y, Taillard ED (1995) Probabilistic diversification and intensification in local search for vehicle routing. J Heuristics 1:147–167

Taillard ED (1993) Parallel iterative search methods for vehicle routing problems. Networks 23:661–672

Tarantilis CD (2005) Solving the vehicle routing problem with adaptive memory programming methodology. Comput Oper Res 32(9):2309–2327

Tarantilis CD, Kiranoudis CT (2002) BoneRoute: an adaptive memory-based method for effective fleet management. Ann Oper Res 115(1):227–241

Tarantilis CD, Kiranoudis CT, Vassiliadis VS (2002) A list based threshold accepting algorithm for the capacitated vehicle routing problem. Int J Comput Math 79(5):537–553

Teo J, Abbass HA (2003) A true annealing approach to the marriage in honey bees optimization algorithm. Int J Comput Intell Appl 3(2):199–211

Teodorovic D, Dell'Orco M (2005) Bee colony optimization—a cooperative learning approach to complex transportation problems. In: Advanced OR and AI methods in transportation, pp 51–60

Toth P, Vigo D (2002) The vehicle routing problem. Monographs on discrete mathematics and applications. Siam, Philadelphia

Wedde HF, Farooq M, Zhang Y (2004) BeeHive: an efficient fault-tolerant routing algorithm inspired by honey bee behavior. In: Dorigo M (ed) Ant colony optimization and swarm intelligence, LNCS 3172. Springer, Berlin, pp 83–94

Yang XS (2005) Engineering optimizations via nature-inspired virtual bee algorithms. In: Yang JM, Alvarez JR (eds) IWINAC 2005, LNCS 3562. Springer-Verlag, Berlin, pp 317–323