Neurocomputing 135 (2014) 98–106

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Urban bicycles renting systems: Modelling and optimization using nature-inspired search methods

Camelia Chira [a,*], Javier Sedano [a], José R. Villar [b], Mónica Cámara [a], Emilio Corchado [c]

[a] Instituto Tecnológico de Castilla y León, c/. López Bravo 70, Pol. Ind. Villalonquéjar, 09001 Burgos, Spain
[b] University of Oviedo, Campus de Viesques s/n, 33204 Gijón, Spain
[c] Departamento de Informtica y Automtica, Universidad de Salamanca, Plaza de la Merced s/n, Salamanca, Spain

ARTICLE INFO

Article history:
Received 19 November 2012
Received in revised form
23 April 2013
Accepted 8 July 2013
Available online 10 January 2014

Keywords:
Urban bicycles renting systems
Vehicle routing problems
Evolutionary algorithms
Ant colony systems

ABSTRACT

Urban Bicycles Renting Systems (UBRS) are becoming a common and useful component in growing modern cities. For an efficient management and support, the UBRS infrastructure requires the optimation of vehicle routes connecting several bicycle base stations and storage centers. In this study, we model this real-world optimization problem as a capacitated Vehicle Routing Problem (VRP) with multiple depots and the simultaneous need for pickup and delivery at each base station location. Based on the VRP model specification, two nature-inspired computational techniques, *evolutionary algorithms* and *ant colony systems*, are presented and their performance in tackling the UBRS problem is investigated. In the evolutionary approach, individuals are encoded as permutations of base stations and then translated to a set of routes subject to the constraints related to vehicle capacity and node demands. In the ant-based approach, ants build complete solutions formed of several subtours servicing a subset of base stations using a single vehicle based on both *apriori* (the attractiveness of a move based on the known distance or other factors) and *aposteriori* (pheromone levels accumulated on visited edges) knowledge. Both algorithms are engaged for the UBRS problem using real data from the cities of Barcelona and Valencia. Computational experiments for several scenarios support a good performance of both population-based search methods. Comparative results indicate that better solutions are obtained on the average by the ant colony system approach for both considered cities.

1. Introduction

Urban Bicycles Renting Systems (UBRS) require the planification of minimal-cost routes for several vehicles used in servicing a set of bicycle base stations distributed across a city. The quality indicators of UBRS functionality include the number of bicycles available in base stations in the morning, the availability level of base station (defined as the number of times encountered during 30 min when a certain base station cannot offer the required service), the duration in time when a base station is full/empty and the percentage of failed attempts from clients to drop off a bicycle at the desired base station (due to the high number of bicycles already attached) [1]. For instance, minimum quality UBRS indicators in the city of Madrid can be 95% availability of bicycles and base stations in the morning, 45 min as the ratio of base station availability, maximum 30 min a base station can stay empty, maximum 60 min a base station can be full and 3% as the percentage of failures. Given these requirements and restrictions, the UBRS management system needs to determine what vehicles are necessary to service the base stations, the routes for each vehicle and the number of bicycles to pick up and/or deliver to each base station.

In this contribution, it is shown how the management of the bicycles renting system can be modeled as a Vehicle Routing Problem (VRP) with certain features. Routing problems are an important class of optimization problems extensively investigated [23,19,28,29,21,9] due to their complexity and large number of real-world applications. In the case of UBRS, we seek to optimize vehicle routes so that the total cost is minimum and all the base stations are serviced by a vehicle (given a set of storage centers, bicycle base stations and vehicles – each with a certain capacity). This is a Capacitated VRP (CVRP) in which customers have simultaneously the need for pickup and delivery and more than one depot (storage center) is available to the fleet of vehicles servicing customers. VRP is a well-known complex combinatorial optimization problem [2,16,26] shown to be NP-hard [17]. This means that VRPs become intractable when the search space is too large due to their combinatorial complexity. There is no exact algorithm able to consistently solve VRP instances with more than 50 customers [8,23]. Therefore, several metaheuristic methods

[*] Corresponding author. Tel.: +34 947 298 471; fax: +34 947 298 091.
E-mail address: camelia.chira@itcl.es (C. Chira).

0925-2312/$ - see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.07.051

including genetic algorithms [4,6,21,22,24,27], tabu search [14,25], cooperation models for optimization agents [3] and ant colony optimization [7,13,20,23,9] have been proposed to find close-to-optimal solutions.

This study presents the development of two population-based search methods to address the UBRS problem modeled as a VRP. The first one is an evolutionary algorithm in which potential solutions are encoded as permutations of base stations. The fitness of an individual is the total cost of all the routes assumed in the solution. The search process relies on simple genetic operators (order crossover and swap mutation) and an elitist strategy. The second method considered for experiments is the Ant Colony System (ACS) metaheuristic [12], which is inspired by the behavior of real ant colonies. Many recent research studies show a good performance of ant-based systems for the VRP [7,20,23]. The ACS for UBRS developed in this study is using as a starting point the Ant System developed for VRP in [7]. Computational experiments are performed for several demand scenarios for UBRS data collected from two Spanish cities: Barcelona and Valencia. Numerical results obtained using the evolutionary and ant-based methods are presented and analyzed. Experiments support a good performance of both methods, which are able to obtain efficient solutions for UBRS. Nevertheless, compared to the evolutionary approach, ACS shows a better performance being able to trigger more cost-efficient vehicle routes.

The structure of the paper is as follows: the real-world problem of urban bicycles renting systems is described and then modeled as a VRP; the evolutionary algorithm and the ant colony system developed to address the UBRS problem are presented; numerical results and comparisons are discussed and conclusions are drawn at the end of this contribution.

2. Problem definition and modelling

The management of UBRS requires the optimization of routes for distributing bicycles across a city using available vehicles of various capacities. This section details this problem and presents its VRP model.

2.1. UBRS description

A UBRS is composed of a set of n base stations (denoted by BS_i where $i=1\ldots n$), each of them allowing a fixed number of bicycles to be chained. These bicycles can be borrowed in a certain BS_i and returned to a different one. For each BS_i, the location, the capacity and the number of bicycles currently chained are known.

There are two storage centers (denoted by SC_0 and SC_{-1}), each of them with a given capacity for storing bicycles. Also, in the storage centers the damaged bicycles can be repaired. The distances between any two nodes (BS_i and BS_j, $i,j=1\ldots n, i\neq j$) as well as the distances between each node (BS_i, $i=1\ldots n$) and the storage centers (SC_0, SC_{-1}) are given. Alternatively, the travel time needed to reach a node from any other might be given in some UBRS case scenarios.

The bicycles are distributed each morning (from the SC to the BS) and gathered at night. When needed, the BS should be visited to adjust the number of bicycles to a certain specific number. For these purposes, there are m vehicles with various capacities. Vehicles can be assigned to an SC and can distribute bicycles to any base station.

The problem in UBRS management is to obtain the best set of routes for distributing the bicycles all along the UBRS. The outcome should include the number and capacity of the vehicles needed in the process, the route of BS nodes that each vehicle should follow (the start and end point of each route is SC_0 or SC_{-1}) and the number of bicycles to be collected and/or dropped from each BS in the route. The cost of each route will be determined as a function of the distance covered by the vehicles.

2.2. Problem representation as VRP

VRP is a well-known NP-hard optimization problem [2,16,17,26] aiming to find the optimal routes for servicing a number of customers using a set of vehicles. Indeed, the UBRS management problem seeks to optimize vehicle routes so that the total cost is minimum and all the base stations are serviced by a vehicle.

Let $G=(V,A)$ be a complete graph with $V=\{-1,0,1,\ldots,n\}$ as a set of nodes and the set of arcs $A=\{(i,j)|i,j\in V, i\neq j\}$. Nodes -1 and 0 represent the storage centers SC_{-1} and SC_0. The remaining node set $V'=V\backslash\{-1,0\}$ corresponds to the n bicycle base stations. Each arc $(i,j)\in A$ is assigned with a cost c_{ij} (i.e. the distance between nodes in this study).

Each node i has a capacity S_i representing the maximum number of bicycles that can be stacked on it. The current number of bicycles assigned to node i is denoted by s_i. Moreover, the desired number of bicycles for each node i is given as d_i. The two storage centers (nodes -1 and 0) do not have any demand for bicycles (it is assumed that $d_{-1}=d_0=0$). Each node $i, i=1\ldots n$ has a demand calculated as d_i-s_i. If this value is positive it means that node i should be serviced by delivering to it a value equal to the difference between d_i and s_i of bicycles, otherwise the vehicle should pick up from the customer node a number of s_i-d_i bicycles. A set of m vehicles $M=\{1,\ldots,m\}$ with different capacities are available to service the customer nodes. The capacity of each vehicle is denoted by Q_k, $\forall k\in M$. A cost associated to each vehicle denoted by C_k is given.

The UBRS management problem aims to find a collection of routes with minimum cost such that (i) each route starts at one of the storage centers (nodes -1 and 0) and ends at the same storage center; (ii) each customer node is visited by exactly one vehicle route; (iii) the sum of demands for the nodes visited in a route by a vehicle does not exceed the capacity of that vehicle; and (iv) the number of vehicles (vehicle routes) is minimized. This problem can be modeled as a VRP [26] with properties from several VRP types: Capacitated VRP (there is a constraint about the capacity of the vehicle servicing a route but the available vehicles are of different capacities), Multiple Depots VRP (the number of storage centers is greater than one), and VRP with Pickup and Delivery (the vehicle visiting a customer node might have to collect or drop off bicycles depending on the current demand of that node).

The mathematical formulation of the UBRS problem modeled as VRP is given below (based on [2]). Flow variables, denoted by $x_{ij}{}^k$ and $y_{ij}{}^k$ for each vehicle k and nodes (i,j), are used to indicate if a vehicle travels between two given base stations:

- $x_{ij}{}^k$ takes the value 1 if vehicle k travels from node i to node j, and 0 otherwise.
- $y_{ij}{}^k$ represents the quantity of bicycles available in vehicle k when it leaves node i to service node j.

Given these notations, the mathematical formulation for the UBRS problem is as follows:

$$\text{minimize} \sum_{k\in M} C_k \sum_{\substack{i\in\{-1,0\}\\ j\in V'}} x_{ij}^k + \sum_{k\in M}\sum_{\substack{i,j\in V\\ i\neq j}} c_{ij}x_{ij}^k \tag{1}$$

subject to

$$\sum_{k\in M}\sum_{i\in V} x_{ij}^k = 1, \quad \forall j\in V' \tag{2}$$

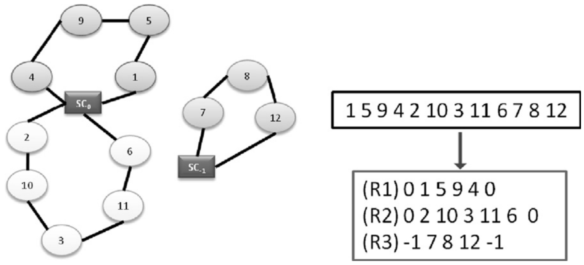

Fig. 1. The permutation-based representation translated into a set of three routes R1, R2 and R3 for an example graph with 12 BS and 2 SC.

$$\sum_{i \in V} x_{ip}^k - \sum_{j \in V} x_{pj}^k = 0, \quad \forall p \in V', \quad \forall k \in M \tag{3}$$

$$y_{ip}^k - y_{pj}^k = d_p - s_p, \quad \forall p \in V', \quad \forall k \in M | x_{ij}^k \neq 0 \tag{4}$$

$$y_{ij}^k \geq d_j - s_j, \quad \forall i,j \in V, i \neq j, \quad \forall k \in M | x_{ij}^k \neq 0, d_j \geq s_j \tag{5}$$

$$y_{ij}^k \leq Q_k - (d_j - s_j), \quad \forall i,j \in V, i \neq j, \quad \forall k \in M | x_{ij}^k \neq 0, d_j < s_j \tag{6}$$

The constraint given in Eq. (2) ensures that any node is visited exactly once and the constraint in Eq. (3) ensures that a vehicle that arrives to node p, also departs from p. Commodity flow constraints from Eq. (4) specify that the difference between the number of bicycles carried by a vehicle before and after visiting a node p equals the demand of base station p. The constraints related to the capacity of the vehicles are given in Eqs. (5) and (6).

Besides the vehicle routes, the initial load of bicycles for each vehicle used to service UBRS is also required as part of the problem solution. It should be noticed that summing up the demands of each base station does not necessarily equal the initial amount of bicycles needed in a vehicle. This is because some of the base stations might require only pickup of bicycles while others require delivery. The order of these requests in the route is important in determining the necessary load of a vehicle when leaving a storage center. Furthermore, each vehicle $k \in M$ needs an initial load that does not exceed neither its own capacity Q_k nor the number of bicycles currently available in the storage center $s_i, i \in \{-1, 0\}$. Also, the number of bicycles in a vehicle when returning to the storage center should not exceed the capacity of that storage center given the current load.

3. An evolutionary approach to UBRS

A genetic algorithm (GA) for solving the VRP as defined in the previous section is developed. GAs [15] represent a powerful search mechanism based on chromosome evolution and operators such as selection, crossover and mutation. The application of GAs to different VRP variant problems has been shown to be successful in many studies [4,22,24,27].

Potential solutions to the problem are encoded as the members of a population which is evolved toward better fitted individuals during a fixed number of generations. The search process is guided by problem-specific genetic operators and the quality of an individual is assessed based on the total route cost (as given in the previous section).

3.1. Representation

An individual incorporates all the routes to be executed and is defined as a permutation of the nodes in V'. The starting and ending nodes of a route should be a storage center. The storage centers (nodes -1 and 0) are not included in the representation as they will be automatically determined. The nearest storage center to the first node in the route is considered as the origin of that route provided there is at least a vehicle with enough capacity in it. Each vehicle returns to the storage center from which it started the route.

The permutation-based encoding of an individual is translated to a set of routes in the following way: (i) the nodes are assigned to a route in the order in which they appear in the permutation; (ii) one of the vehicles available at the nearest storage center (used as start and end point of the route) is randomly selected to service the route subject to the constraints regarding vehicle capacity specified in the previous section; and (iii) a new route for another vehicle is created when the current node in the permutation cannot be serviced by the vehicle previously selected.

Let us consider an example of an UBRS graph with 14 nodes: SC_0 and SC_{-1} represent the storage centers and nodes labeled from 1 to 12 represent the base stations. Fig. 1 presents a possible chromosome value of 1 5 9 4 2 10 3 11 6 7 8 12, representing a permutation of integers from 1 to 12 (the nodes representing SC are not included in the chromosome encoding). The translation process aims to find the corresponding set of routes and starts by finding the SC closest to first node in the route (node 1 in our example) – let us assume SC_0 is the nearest SC to node 1. A vehicle available in SC_0 is randomly selected (with a bias towards higher capacity vehicles). When a node cannot be serviced by the same vehicle, the route is split (adding the same end node as the start SC) and a new route is started. Depending on the demands of each node and the available vehicles, one possible translation of the chromosome value from Fig. 1 to a set of three routes is 0 1 5 9 4 2 0, 0 2 10 3 11 6 0 and -1 7 8 12 -1.

3.2. The fitness function and genetic operators

After the translation process, each individual is evaluated using Eq. (1). This function ensures the minimization of both the total cost of routes and the number of vehicles used. The genetic operators are those commonly engaged for permutation-based encoding: order crossover and swap mutation. Order crossover selects a substring from the first parent at random. This substring is copied in the offspring at the same positions. The rest of required nodes (in order to keep the permutation valid) are taken from the second parent in the corresponding order. Swap mutation picks two alleles at random and swaps their positions preserving most of the adjacency information.

3.3. The evolutionary algorithm

The proposed GA uses a fixed size population initially randomly generated. An elite subpopulation is selected automatically for next generation. Roulette-wheel selection is used to determine the rest of this intermediary population. Individuals from the current population are therefore selected proportionally to their fitness and forming in this way the basis for the next generation.

In the next step, crossover is applied with a certain probability and the generated offspring replaces the first parent if it has a better fitness. For each individual (which is not part of the elite), a mate is selected using binary tournament selection (i.e. two individuals are randomly picked from the population and the one with a better fitness is returned). It should be noted that the entire population is used to select a mate so that any individual including the elite is considered in this process. Furthermore, only the first parent (which is not an elite individual) can be replaced by the offspring in an asynchronous way (replacement occurs during the same generation and the newly generated offspring might be selected as a mate within the same generation).

Mutation is applied to each individual (except the elite individuals) from the intermediary population with a probability (given as a parameter). The offspring generated based on swap mutation replaces the parent if better fitted.

The population obtained by these genetic modifications enters a new search process in the next generation being evaluated against the fitness function. The algorithm stops after a specified maximum number of generations and the best fitted individual is returned as a potential solution to the given problem.

4. Ant colony optimization for UBRS

Ant Colony Optimization – ACO [5,12] is a nature-inspired metaheuristic which replicates the behavior of insects to the search space. It has been successfully applied to travelling salesman and vehicle routing combinatorial optimization problems [7,13,20,23,12]. Ants construct a solution visiting a list of nodes in a graph and laying a pheromone trail on the followed path. The decision regarding the movement along an edge to the next node to visit depends on its attractiveness and the corresponding amount of pheromone. Stronger pheromone trails are preferred and the most promising tours receive a greater pheromone trail in time.

There are several ACO implementations, e.g. Ant System, Ant-Q, and Ant Colony System [11,12,23], mainly differentiated by the way in which the pheromone is used and updated. The first ACO implementation was the Ant System which uses a random-proportional rule to probabilistically select next node to visit and pheromone update takes place at the end of an iteration. Ant System was significantly improved by Ant Colony System (ACS) [11,12] which introduced the pseudo-random-proportional transition rule as well as local and global pheromone update.

One of the first ACO approaches to VRP is presented in [7] for the capacitated version of the problem. Moreover, VRP with Time Windows (VRPTW) is addressed using a multi ant colony system in [13] and the time dependent VRPTW in [10] while the dynamic VRP has an ACO-inspired solution presented in [20].

To address the UBRS problem modeled as VRP, we develop in this research an ACS solution using as a starting point the Ant System proposed in [7] for the Capacitated VRP. The Ant System for VRP described by Bullnheimer et al. [7] is upgraded to an ACS and all the restrictions related to UBRS requirements are considered when building ant solutions. It should be noted that in the ACS approach presented in this study we do not use any elitist ants to build a solution as described in [7].

The number of ants is denoted by m and the number of base stations is denoted by n. Each ant builds a solution individually which is evaluated using the cost function in Eq. (1). Initially, each ant is placed in one of the BS at random and starts building the subtours part of a complete solution. The starting node for each subtour is one of the storage centers SC_0 or SC_{-1} selected in the same way as for the evolutionary approach described in the previous section (one of the vehicles available at the SC nearest the first node in the subtour considering also the restrictions defined in section two).

Let J^* denote the cost (calculated according to Eq. (1)) of an UBRS solution produced by an ant ignoring any pheromone information (similar to a probabilistic nearest neighbor solution). The pheromone level for each edge is initialized to a value denoted by τ_0 calculated as follows:

$$\tau_0 = \frac{1}{n \cdot J^*} \tag{7}$$

Ants select the next BS to visit in a tour based on two factors [12,23]: (i) the attractiveness of a move from node i to node j, denoted by η_{ij} (a priori knowledge); and (ii) the pheromone level τ_{ij} on the edge (a posteriori knowledge). The value of η_{ij} is normally chosen to be the inverse of the cost associated with traversing the edge from i to j. In this work, we set the value of η_{ij} according to the parametrical saving function [7] as follows:

$$\eta_{ij} = c_{i0} + c_{oj} - g \cdot c_{ij} + f \cdot |c_{i0} - c_{0j}|, \tag{8}$$

where node 0 refers to the SC selected to service the current subtour, while f and g are parameters (good settings are $f = g = 2$ [7,23], also used in the current paper).

Let q_0 be a constant such that $0 \le q_0 \le 1$ and q a random variable uniformly distributed over [0, 1]. From a current node i, an ant k will choose the next node j according to the pseudo-random-proportional rule given below:

$$\begin{cases} \arg\max \tau_{ij} \cdot \eta_{ij} & \text{if } q \le q_0 \\ p_{ij}^k & \text{otherwise} \end{cases} \tag{9}$$

If $q > q_0$, the next node j to visit is chosen using the Ant System random-proportional rule:

$$p_{ij}^k = \begin{cases} \dfrac{\tau_{ij}^\alpha \cdot \eta_{ij}^\beta}{\sum_{h \in \Omega} \tau_{ih}^\alpha \cdot \eta_{ih}^\beta} & \text{if } j \in \Omega \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where α and β are parameters that give weight to the influence of trails and visibility, and Ω represents the set of nodes which are feasible to be visited from i. A node is considered feasible if all the restrictions about the demand of the next BS denoted by j and vehicle capacity are satisfied.

After an ant chooses edge $(i - j)$ as part of its solution, a local pheromone update takes place and the trail τ_{ij} is modified according to the local update rule:

$$\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \tau_0, \tag{11}$$

where ρ is a constant.

At the end of each iteration, the global update rule allows the modification of pheromone levels only for the edges that belong to the best tour detected. Let C^* be the cost value (Formula (1)) of the best solution found so far. Only the edges $(i - j)$ that belong to the best solution tour undergo the ACS global pheromone update:

$$\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \frac{1}{C^*} \tag{12}$$

ACS facilitates a balance between the exploration (probability $1 - q_0$) and exploitation (probability q_0) of the search space. After a maximum number of iterations, the ACS algorithm returns the best tour found. This tour is defined by a set of subtours with their corresponding BS to be serviced, one SC as the end and start location and the vehicle used in each subtour.

5. Computational experiments and results

Computational experiments are performed for UBRS real data from the cities of Barcelona and Valencia. The results of the evolutionary algorithm and ant colony system are presented and compared in this section.

5.1. Dataset description

The datasets for the current experiments are created based on real data from two major cities in Spain, as already mentioned. Table 1 presents some details about these datasets including the number of base stations considered (randomly selected from the total number of real stations available in Barcelona and Valencia respectively) and their capacities, the number and capacity of available vehicles. The two storage centers SC_0 and SC_{-1} can store among 100 and 200 bicycles (the exact capacities are given by S_0 and S_{-1}). The number of vehicles available for UBRS management is $m = 16$ vehicles, 8 of them with a capacity of 11 bicycles and the other 8 with a capacity of 20 bicycles which can be extended up to 40 if a trailer is used. When a vehicle is selected from a storage center to service a route, those of higher capacity are preferred and a trailer is randomly added.

The real locations of the BS in the city map are used to compute the distance between any two nodes. It should be noted that the resulted distance matrix is asymmetric as the distance from a node BS_i to a node BS_j might be different than the distance from BS_j to BS_i ($\forall i, j$). Fig. 2 presents the location map with the distribution of nodes for each dataset. The x and y axis from Fig. 2 represent coordinates so that the real location in space and the distance between any two BS is depicted. Base stations are represented as dots or stars depending on their capacity (25 or 30) and storage centers are represented as circles.

Ten scenarios have been randomly generated for each dataset to consider different potential situations of the real UBRS demands. These scenarios have different values for the current number of bicycles s_i and the desired number of bicycles d_i in each $BS_i, i = 1 \ldots n$. An example of such a scenario is depicted in Fig. 3: the nodes (BS_i) are placed on the x axis and the demand (d_i) of each node is given on the y axis. The demand can be positive (meaning a number of d_i bicycles are needed in BS_i) or negative (suggesting a number of bicycles for pickup).

5.2. GA results

The following parameters have been set for the GA: the population size is 100, the number of generations is 200 and the elite size is set to 10% of the population. The population size is chosen based on some preliminary experiments performed for 30, 100 and 500 individuals which show an important improvement in solution fitness when increasing the population size from 30 to 100 but a minor improvement (particularly when reported to the higher computational cost) for 500 population size. Furthermore, the experiments reported in this section are based on a crossover and mutation rate of 0.8. These values are based on preliminary results obtained for different pairs of crossover and mutation probabilities which consider all possible combinations of the following values: 0.2, 0.4, 0.6 and 0.8. Fig. 4 presents the average fitness obtained over 10 GA runs with different crossover and mutation probabilities. It can be noticed that the performance improves with higher crossover rates and the best results are obtained for crossover rates 0.6 and 0.8 for all 10 scenarios in both datasets. Mutation rate of 0.2 triggers the worst performance for any crossover rate while the higher values of 0.6 and 0.8 clearly improve the results. As swap mutation only affects two positions in the solution (out of more than 100 BS), it is not surprising that a higher mutation probability enhances the results of the search process.

Table 1
Datasets details.

Data	Barcelona	Valencia
Total number of BS	112	110
BS with capacity 25	75	74
BS with capacity 30	37	36
Capacity of SC_0 and SC_{-1}	[100,200]	[100,200]
Vehicles	16	16
Vehicles with $Q = 11$	8	8
Vehicles with $Q = 20/40$	8	8

The GA with the selected parameter setting is executed 30 times for each scenario and dataset. The algorithm performance is reliable producing similar results in all the runs considered. This robust behavior is emphasized in Fig. 5 which presents the fitness improvement from one generation to another averaged over the considered runs in the ten scenarios for each dataset. The results show that the major solution improvements occur in fact in the first 60 generations of the GA and a much less significant fitness increase occurs from generation 100 to 200.

5.3. ACS results

For the ACS approach, the following parameter setting is used: $q_0 = 0.9$, $\alpha = 1$, $\beta = 2$, $\rho = 0.1$, $f = g = 2$, and the maximum number of iterations $I_{max} = 2 \cdot n$ (a similar setting is suggested by existing studies [23]). The number m of ants considered is 10. The results are based on 30 runs of the algorithm. Each ant builds a complete valid solution, formed of one or more vehicle tours, which does not need any other translation process. The results presented here use the solution cost to evaluate an ant complete tour and this value is calculated using the same Eq. (1) as in the fitness function of the GA.

Fig. 6 presents the average of the best solution cost for each scenario depicted each iteration. The iteration number corresponds to the x axis and the maximum value of this number is 224 for the Barcelona dataset and 220 for Valencia. It can be noticed that the highest improvement in the solution cost occurs in the first 40–60 iterations of the ACS. The best solution remains the same then for several iterations until an ant is able to detect a better solution.

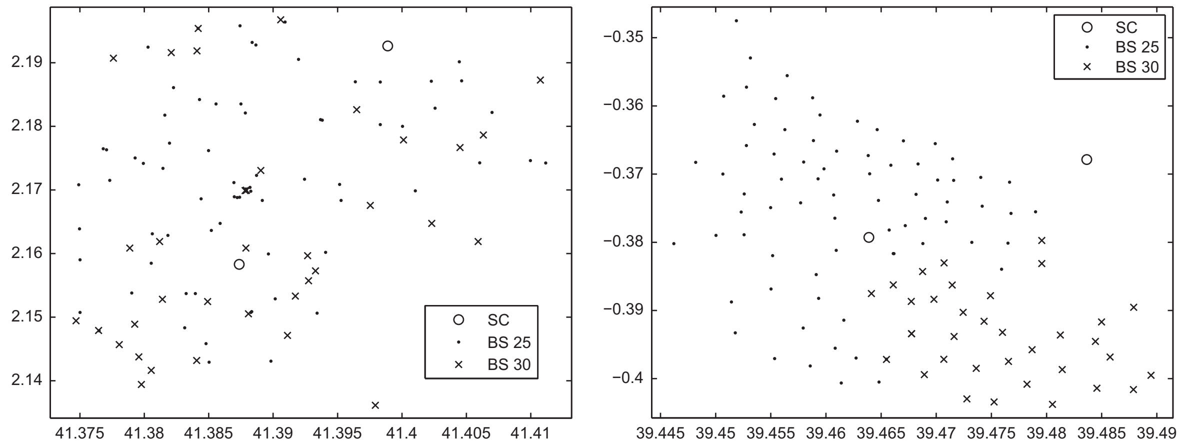

Fig. 2. Location map for Barcelona (left) and Valencia (right) datasets.

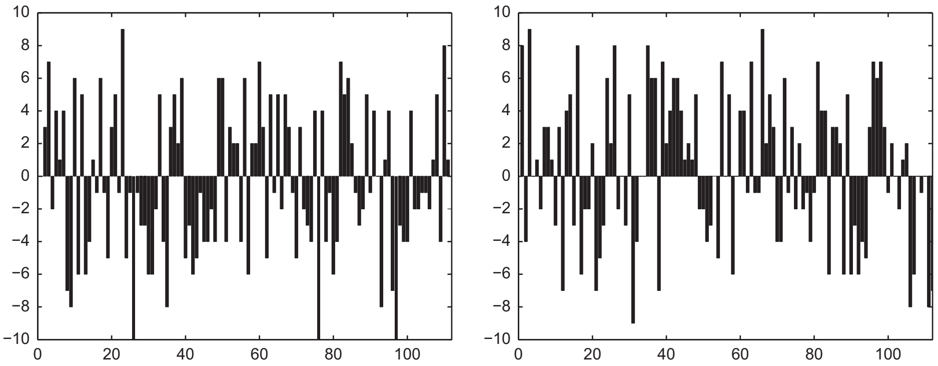

Fig. 3. Case study example scenarios: the required service in each BS on the *x* axis is depicted (negative value suggests pickup of bicycles is required and positive value denotes bicycles that have to be delivered to BS).

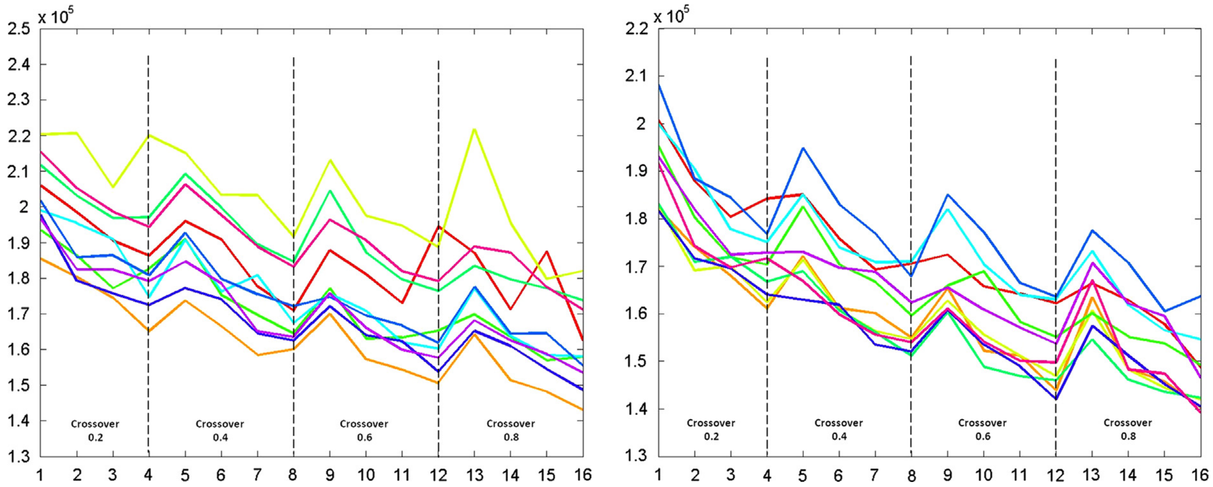

Fig. 4. Average solution fitness over 10 runs of the evolutionary algorithm for Barcelona (left) and Valencia (right) datasets. The *x* axis points 1–16 correspond to pair values of (crossover rate, mutation rate) ranging from (0.2,0.2) in point 1, (0.2,0.4) in point 2 to (0.8,0.6) in point 15 and (0.8,0.8) in point 16. The *y* axis represents the solution fitness. Each line depicts the results for one scenario.

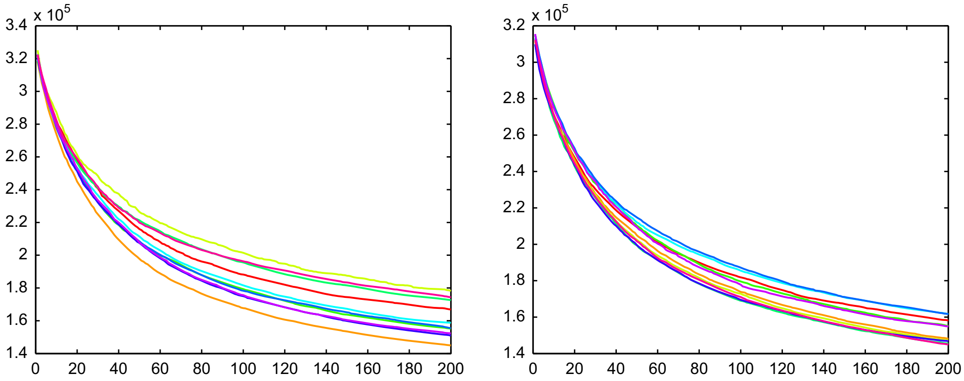

Fig. 5. Average fitness improvement obtained by GA for all considered scenarios (each scenario is depicted by one line) for Barcelona (left) and Valencia (right) over 30 runs. The number of generations is depicted on the *x* axis and the solution fitness is given on the *y* axis.

5.4. *Comparative analysis*

The boxplots of the 30 experiment runs for each of the scenarios are shown in Fig. 7 for the GA and Fig. 8 for the ACS approach. The numerical results obtained for each scenario are given in Table 2 which presents the best and average solution in 30 runs as well as the standard deviation reported by GA and ACS in all experiments. For both methods, the results from all the runs are quite similar in each scenario. For the evolutionary algorithm, only one scenario for each dataset shows a reduced sparsity of results probably induced by premature convergence. For the ACS results, a higher presence of outliers is observed compared to the evolutionary results: 3 scenarios in the Barcelona study case and 4 scenarios in the Valencia dataset. Furthermore, an upper skewness pattern is observed for the ACS boxplot which is an indication of the presence of most solutions in the upper part of the scale.

Both methods are able to provide a solution which includes all necessary routes for servicing all base stations. Table 3 presents the best GA and ACS solutions obtained in 30 runs for each of the scenarios considered for the Barcelona dataset. For each route, the vehicles needed, the base stations visited and the operations performed in each of them are determined. Solutions

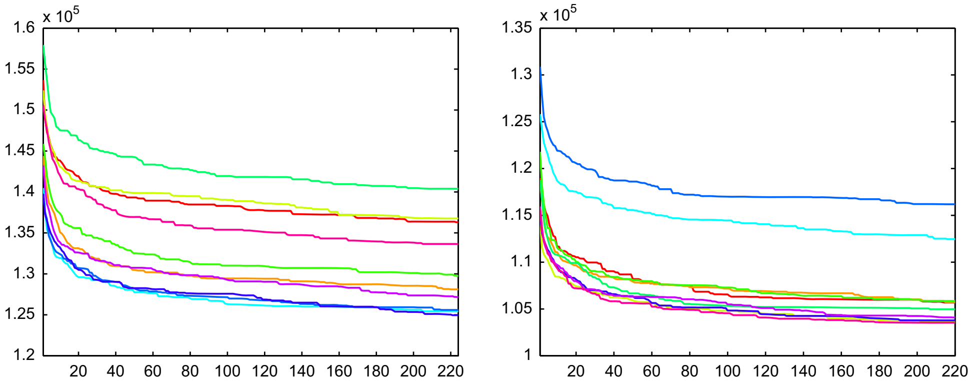

Fig. 6. Best ACS solution averaged over 30 runs over all iterations for all considered scenarios (each scenario is depicted by one line) for Barcelona (left) and Valencia (right). The number of iterations is depicted on the *x* axis and the solution fitness is given on the *y* axis.

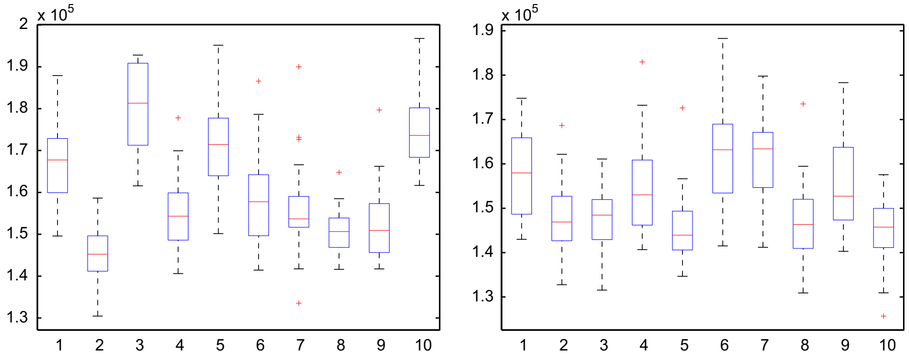

Fig. 7. Boxplot results of the evolutionary algorithm for each scenario in the Barcelona (left) and Valencia (right) study cases. The *x* axis gives the scenario number corresponding to each boxplot and the *y* axis represents the solution fitness.

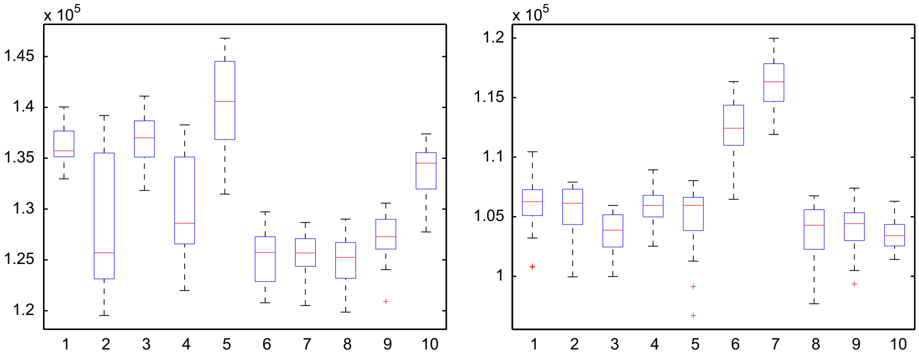

Fig. 8. Boxplot results of the ACS algorithm for each scenario in the Barcelona (left) and Valencia (right) study cases. The *x* axis gives the scenario number corresponding to each boxplot and the *y* axis represents the solution fitness.

are described in Table 3 in terms of fitness (cost), number of nodes in the route, the maximum capacity of the vehicle selected for the route (*Q*) and the initial load of bicycles required in the vehicle (*y*). The results support a better performance of the ACS method in terms of the total route cost which is lower than that of the evolutionary solution for all scenarios. This comparative performance is further supported by Fig. 9 which presents the results of GA and ACS for both Barcelona and Valencia datasets.

As shown in Table 3, the solution cost for some scenarios (e.g. Scenario 7 and Scenario 8) is not the same for both GA and ACS although the number of BS used in the tour is the same. This is due to the different order of BS in the two solutions, which also affects the capacity needed for a vehicle and the number of bicycles *y* required as the initial vehicle load.

Indeed, an important difference between the GA and ACS approaches refers to the way in which solutions are constructed. In the case of GA, the creation of a new solution starts from a permutation of BS which is then translated to a valid set of vehicle routes to correctly service the current UBRS needs. The ACS approach builds a solution by adding a new BS each step to the current vehicle tour (which is in fact a part of a whole tour solution). The solutions evolved by GA rely to a high degree on the order of BS given by the initial permutation while ants choose the best next BS based on an intermediary solution. This difference might explain the comparative performance of the two methods.

6. Conclusions and future work

An evolutionary algorithm and an ant colony system model have been designed and implemented for the management of urban bicycle renting systems. An important contribution of this

Table 2
Results of GA and ACS for all considered problems (10 scenarios in each city, B for Barcelona and V for Valencia) for 30 independent runs: best solution cost, average performance and standard deviation.

City-	GA			ACS		
Sc.	Best	Avg.	Std. Dev.	Best	Avg.	Std. Dev.
B-1	149,581	166905.4	8982.1	132,972	136301.2	1854.9
B-2	130,459	144925.8	6782.06	119,528	128103.6	6538.11
B-3	161,568	179792.76	10418.81	131,828	136767.9	2370.58
B-4	140,611	155172.2	8979.82	121,974	129745.46	4877.77
B-5	150,159	172635.16	10639.8	131,470	140400	4251.41
B-6	141,432	158743.5	10771.49	120,780	125438.6	2546.84
B-7	133,547	155497.5	11180.36	120,507	125600.6	1982.02
B-8	141,623	150928.96	5243.90	119,855	124955.6	2295.72
B-9	141,723	152314.7	8426.24	120,905	127149.5	2156.43
B-10	161,655	174381.23	8321.13	127,751	133650.43	2682.42
V-1	143,000	158075.5	9660.01	100,761	105682.03	2406.16
V-2	132,771	148069.6	8272.31	99,952	105721.73	1892.82
V-3	131,551	147108.33	7370.62	99,971	103599.63	1820.01
V-4	140,682	155225.03	11050.44	102,521	105856.86	1543.24
V-5	134,661	145495.2	7505.26	96,704	104962.93	2613.24
V-6	141,512	161765.93	11180.68	106,459	112463	2312.40
V-7	141,202	161700.8	8327.95	111,904	116185.06	2055.59
V-8	130,900	146772.46	9124.48	97,692	103779.63	2180.76
V-9	140,280	154909.6	9269.06	99,344	104082.46	1931.91
V-10	125,692	144976.23	6871.59	101,409	103534.06	1216.68

Table 3
Best solutions obtained for each considered scenario by GA and ACS in the Barcelona case study.

Sc.	GA cost	GA solution	ACS cost	ACS solution
1	149,581	*1*: 108 BS, Q 40, y 6 *2*: 4 BS, Q 20, y 0	132,972	*1*: 111 BS, Q 40, y 2 *2*: 1 BS, Q 20, y 0
2	130,459	*1*: 82 BS, Q 20, y 11 *2*: 30 BS, Q 20, y 9	119,528	*1*: 112 BS, Q 20, y 19
3	161,568	*1*: 24 BS, Q 40, y 17 *2*: 25 BS, Q 40, y 23 *3*: 63 BS, Q 40, y 27	131,828	*1*: 107 BS, Q 40, y 40 *2*: 5 BS, Q 40, y 30
4	140,611	*1*: 112 BS, Q 40, y 7	121,974	*1*: 112 BS, Q 40, y 0
5	150,159	*1*: 38 BS, Q 40, y 12 *2*: 61 BS, Q 40, y 10 *2*: 13 BS, Q 20, y 0	131,470	*1*: 110 BS, Q 40, y 9 *2*: 2 BS, Q 20, y 0
6	141,432	*1*: 112 BS, Q 40, y 21	120,780	*1*: 112 BS, Q 40, y 38
7	133,547	*1*: 112 BS, Q 40, y 23	120,507	*1*: 112 BS, Q 40, y 39
8	141,623	*1*: 112 BS, Q 40, y 14	119,855	*1*: 112 BS, Q 20, y 19
9	141,723	*1*: 112 BS, Q 40, y 11	120,905	*1*: 112 BS, Q 40, y 27
10	161,655	*1*: 91 BS, Q 40, y 28 *2*: 18 BS, Q 20, y 19 *3*: 3 BS, Q 20, y 9	127,751	*1*: 109 BS, Q 40, y 39 *2*: 3 BS, Q 20, y 17

study refers to the modelling of the real-world problem of UBRS management as a vehicle routing problem. It has been shown that the considered problem is a capacitated VRP with multiple depots and the simultaneous need for pickup and delivery at each base station location. The proposed nature-inspired techniques rely on the mathematical formulation of the problem. Computational experiments show how the UBRS management problem is addressed using the GA and ACS approaches indicating the solution components and quality. In terms of performance, the ACS method generates better solutions compared to the GA but requires more computational resources.

The performance of both methods can be significantly enhanced by integrating a local search mechanism for a good exploitation of search space. This extension represents the focus of our future work. Furthermore, the GA could benefit from a diversification stage periodically engaged during the population evolution to replace similar individuals by newly generated genetic material. It is expected that the maintenance of a diverse population will avoid premature convergence resulting in better performance. The ant-based approach will be further studied by tuning the ACS parameters and integrating a local search phase to better exploit ant solutions. Future work also focuses on the design and implementation of other nature-inspired metaheuristics such as Particle Swarm Optimization [18] and their comparative performance in relation to the methods presented in the current paper for the UBRS problem.

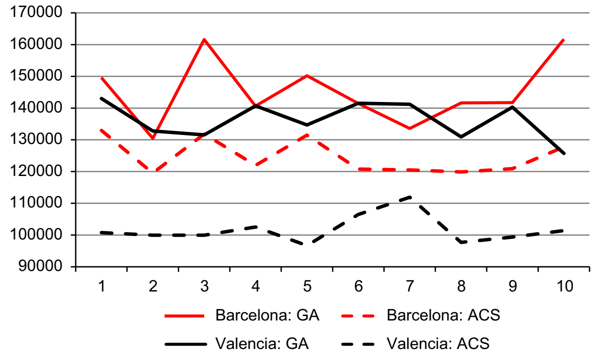

Fig. 9. Best solution cost (*y* axis) for each of the 10 scenarios (*x* axis) in the Barcelona and Valencia datasets detected by GA and ACS.

Acknowledgments

This research is funded by the Spanish Ministry of Economy and Competitiveness, under project CDTI IDI-2012012, the Junta de Castilla y León CCTT/10/BU/0002 and the Ministry of Economy and Competitiviness project TIN2011-24302. This research is also partially supported through projects of the Spanish Ministry of Economy and Competitiveness with ref: TIN2010-21272-C02-01 (funded by the European Regional Development Fund), and SA405A12-2 from Junta de Castilla y León.

References

[1] Announcement of the Municipal Transport Company of Madrid, S. A. For the Design, Implementation, Exploitation and Maintenance of a Public Bicycle Model in Madrid (MyBICI), BOE no. 10, Spain, 2010.
[2] R. Baldacci, P. Toth, D. Vigo, Recent advances in vehicle routing exact algorithms, 4OR: Q. J. Oper. Res. 5 (4) (2007) 269–298.
[3] D. Barbucha, Search modes for the cooperative multi-agent system solving the vehicle routing problem, Neurocomputing 88 (2012) 13–23.
[4] B.M. Baker, M.A. Ayechew, A genetic algorithm for the vehicle routing problem, Comput. Oper. Res. 30 (2003) 787–800.
[5] E. Bonabeau, M. Dorigo, G. Theraulaz, Inspiration for optimization from social insect behaviour, Nature 406 (2000) 39–42.
[6] O. Braysy, W. Dullaert, M. Gendreau, Evolutionary algorithms for the vehicle routing problem with time windows, J. Heuristics 10 (2004) 587–611.
[7] B. Bullnheimer, R.F. Hartl, C. Struss, An improved ant system algorithm for the vehicle routing problem, Ann. Oper. Res. 89 (1999) 319–328.
[8] J.-F. Cordeau, G. Laporte, Tabu search heuristics for the vehicle routing problem, Oper. Res./Comput. Sci. Interfac. Ser: Springer US 30 (2005) 145–163.
[9] Q. Ding, X. Hu, L. Sun, Y. Wang, An improved ant colony optimization and its application to vehicle routing problem with time windows, Neurocomputing 98 (2012) 101–107.
[10] A.V. Donati, R. Montemanni, N. Casagrande, A.E. Rizzoli, L.M. Gambardella, Time dependent vehicle routing problem with a multi ant colony system, Eur. J. Oper. Res. 185 (3) (2008) 1174–1191.
[11] M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Trans. Evol. Comput. 1 (1) (1997) 53–66.
[12] M. Dorigo, Ant colony optimization, Scholarpedia 2 (3) (2007) 1461.
[13] L.M. Gambardella, E. Taillard, G. Agazzi, MACS-VRPTW: a multiple ant colony system for vehicle routing problems with time windows, in: D. Corne, M. Dorigo, F. Glover (Eds.), New Ideas in Optimization, McGraw-Hill, London, UK, 1999, pp. 63–76.
[14] M. Gendreau, A. Hertz, G. Laporte, A tabu search heuristic for the vehicle routing problem, Manag. Sci. 40 (10) (1994) 1276–1290.
[15] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, MA, 1989.

[16] G. Laporte, The Vehicle Routing Problem: an overview of exact and approximate algorithms, Eur. J. Oper. Res. 59 (1992) 345–358.
[17] L. Bodin, B. Golde, Classification in vehicle routing and scheduling, Networks 11 (1981) 97–108.
[18] H. Liu, A. Abraham, An hybrid fuzzy variable neighborhood particle swarm optimization algorithm for solving quadratic assignment problems, J. Univers. Comput. Sci. 13 (7) (2007) 1032–1054.
[19] T.A.S. Masutti, L.N. de Castro, Neuro-immune approach to solve routing problems, Neurocomputing 72 (2009) 2189–2197.
[20] R. Montemanni, L.M. Gambardella, A.E. Rizzoli, A.V. Donati, Ant colony system for a dynamic vehicle routing problem, J. Comb. Optim. 10 (2005) 327–343.
[21] P.C. Pop, O. Matei, C. Pop Sitar, An improved hybrid algorithm for solving the generalized vehicle routing problem, Neurocomputing 109 (2013) 76–83.
[22] C. Prins, A simple and effective evolutionary algorithm for the vehicle routing problem, Comput. Oper. Res. 31 (2004) 1985–2002.
[23] A.E. Rizzoli, R. Montemanni, E. Lucibello, L.M. Gambardella, Ant Colony Optimisation for real world vehicle routing problems: from theory to applications, Swarm Intell. 1 (2) (2007) 135–151.
[24] H.G. Santos, L.S. Ochi, E.H. Marinho, L.M.A. Drummond, Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem, Neurocomputing 70 (1–3) (2006) 70–77.
[25] E.D. Taillard, P. Badeau, M. Gendreau, F. Guertin, J.Y. Potvin, A tabu search heuristic for the vehicle routing problem with soft time windows, Transp. Sci. 31 (1997) 170–186.
[26] P. Toth, D. Vigo, The Vehicle Routing Problem, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.
[27] C.-H. Wang, J.-Z. Lu, A hybrid genetic algorithm that optimizes capacitated vehicle routing problems, Expert Syst. Appl. 36 (2009) 2921–2936.
[28] X. Wang, H. Qu, Z. Yi, A modified pulse coupled neural network for shortest-path problem, Neurocomputing 72 (2009) 3028–3033.
[29] J. Zhang, X. Feng, B. Zhou, D. Ren, An overall-regional competitive self-organizing map neural network for the Euclidean traveling salesman problem, Neurocomputing 89 (2012) 1–11.

Camelia Chira received the M.Sc. (2002) and Ph.D. (2005) degrees from Galway – Mayo Institute of Technology, Ireland in the area of agent-based systems for distributed collaborative design environments. Since 2006, Camelia is a researcher at the Department of Computer Science, Babes-Bolyai University, Romania. Currently, she is also working with the Artificial Intelligence and Applied Electronics group from Instituto Tecnológico de Castilla y Leóon, Spain. Her main research interests include Nature-inspired computing, Complex systems and networks, Multi-agent systems and Bioinformatics.

Javier Sedano is an expert in the development of electronic systems (hardware), industrial project, production-systems acquisition and control systems, as well as in the design of connectionless models for the identification and modelling of dynamic systems. He is the head of the Artificial Intelligence and Applied Electronics group at the Instituto Tecnológico de Castilla y Leóon. He is also part of the Applied Computational Intelligence Group at the University of Burgos and has spent several years working on projects and publications related to artificial intelligence and system modelling. He has collaborated in the organization of international scientific conferences, program committees and organizations. He is a member of the IEEE Systems, Man & Cybernetics Chapter Society Spanish Chapter and has several international publications as well as records of software and industrial patents.

Josée R. Villar obtained his Engineering degree at the University of Oviedo (1992) and his Ph.D. at the University of León (2002). He had worked for several engineering companies between 1992 and 1998. In 1998 he became an Assistant Professor with the Electric and Electronic Department at the University of León. In 2004 he became an Assistant Professor with the Computer Science Department at the University of Oviedo. Since 2008, he is an Associate Professor with this department. He had participated in several research projects and contracts with public and private funding and had published contributions in indexed international journals and ranked conferences as well as several book chapters. He is a member of the IEEE Systems, Man & Cybernetics Chapter Society Spanish Chapter. His research interests are focused in Genetic Fuzzy Systems, Meta-heuristics and Low Quality Data processing.

Móonica Cámara obtained her Computer Science Engineering Degree at the University of Burgos in 2011. Since 2010 she is working with the Artificial Intelligence and Applied Electronics group at the Instituto Tecnolóogico de Castilla y León. She is currently also engaged in a Master program in the area of Computer Engineering at the University of Burgos, Spain. Her research interests are focused in Genetic Algorithms, Meta-heuristics and Low Quality Data processing.

Emilio Corchado is an Associate Professor of Computer Science at the University of Salamanca (BISITE Research Group and GICAP Research Group). He is chair of the IEEE Systems, Man and Cybernetics-Spanish Chapter; member of the IEEE Spanish Section Board and member of the IEEE R8 PA subcommittee. He is currently or has previously been a member of various European networks for Excellence as Nature-inspired Smart Information Systems (NiSIS) and COST Action IC0806: Intelligent Monitoring, Control and Security of Critical Infrastructure Systems (IntelliCIS). He has been reviewer and evaluator for the European Commission for the 7FP since 2007. His research interests include neural networks, with a particular focus on exploratory projection pursuit, maximum likelihood Hebbian learning, self-organising maps, multiple classifier systems and Hybrid Artiticial Systems and its applications to industrial and environmental problems. He has published over 200 peer-reviewed articles in a range of topics including modelling of industrial processes, knowledge management and risk analysis, intrusion detection systems, food industry, artificial vision, and so on. He is member of the Editorial Boards of the International Journal of Computational Intelligence and Applications (IJCIA) and also of the International Journal of Reasoning-based Intelligent Systems (IJRIS). He is a guest editor of several special issues of reputed journals such as Information Science (Elsevier), Knowledge and Information Systems (Springer), Neurocomputing (Elsevier), International Journal of Neural Systems (World Scientific), Logic Journal of the IGPL (Oxford Journals), Neural Networks World (Institute of Computer Science AS CR and Faculty of Transportation Sciences), Expert Systems (Wiley-Blackwell). He is general chair for the following conferences: International Conference on Hybrid Artificial Intelligence Systems (HAIS), International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO) and International Conference on Computational Intelligence in Security for Information Systems (CISIS). He has been visiting professor at universities in Poland and Czech Republic and given plenary talks at several international conferences as HIS/IAS 2011.