



Optimal allocation of public parking spots in a smart city: problem characterisation and first algorithms

Javier Arellano-Verdejo, Federico Alonso-Pecina, Enrique Alba & Adolfo Guzmán Arenas

To cite this article: Javier Arellano-Verdejo, Federico Alonso-Pecina, Enrique Alba & Adolfo Guzmán Arenas (2019): Optimal allocation of public parking spots in a smart city: problem characterisation and first algorithms, Journal of Experimental & Theoretical Artificial Intelligence, DOI: [10.1080/0952813X.2019.1591522](https://doi.org/10.1080/0952813X.2019.1591522)

To link to this article: <https://doi.org/10.1080/0952813X.2019.1591522>



Published online: 03 Apr 2019.





Submit your article to this journal [↗](#)



View Crossmark data [↗](#)



Optimal allocation of public parking spots in a smart city: problem characterisation and first algorithms

Javier Arellano-Verdejo ^a, Federico Alonso-Pecina ^b, Enrique Alba^c
and Adolfo Guzmán Arenas^d

^aEstación para la Recepción de Información Satelital ERIS, Colegio de la Frontera Sur ECOSUR, Chetumal, México;

^bUniversidad Autónoma del Estado de Morelos, Faculty of Administration, Accounting and Informatics,

Cuernavaca, Mexico; ^cDepartment of Languages and Computer Science, Universidad de Málaga, Málaga, Spain;

^dCentro de Investigación en Computación, Instituto Politécnico Nacional, Ciudad de México, Mexico

ABSTRACT

Having a mechanism to mathematically model the problem of the optimal allocation of parking spots within cities could bring great benefits to society. According to the International Parking Institute, about 38% of the cars circulating throughout a city are looking for available parking spots, leading to increased pollution and subsequent health problems, as well as economic losses due to wasted man-hours. In the work presented here, a new mathematical model describing the problem of the optimal allocation of parking spots is proposed, along with an evolutionary algorithm to demonstrate how this model can be used in practice. A simulated annealing algorithm was implemented to test the effectiveness of this approach. The proposed strategy will allow users to find parking more quickly and easily, as well as lead to new services for the hot-topic of smart mobility. For the definition of the problem, a real map of the city of Malaga, Spain, was used along with Sumo software to carry out the simulations. The results clearly demonstrated that the proposed mechanism is capable of minimising the global cost of parking, implying a direct benefit for users.

ARTICLE HISTORY

Received 18 May 2018

Accepted 3 March 2019

KEYWORDS

Parking; smart cities; evolutionary algorithms; NP problem

Introduction

Vehicular parking and traffic have been two major concerns in modern cities since the beginning of the twentieth century (Falcocchio & Levinson, 2015). Over time, they have changed in character, geography, and scale. Congestion and parking are intertwined with one another; looking for a parking space impairs local circulation and creates delays. This leads to increases in fuel use and costs, additional air pollution and greenhouse gas emissions, losses in economic opportunities, and a general detriment to the quality of life in a city (Künzli et al., 2000).

Motivation

The causes of traffic congestion tend to be complex, and lack of parking is not the sole reason; it is, however, a major contributing factor. In 2015, the International Parking Institute estimated that about 38% of the cars circulating in a city are looking for parking spaces (Parking, 2016). In central areas of large cities, drivers can spend upwards of 20 minutes looking for a parking spot. Moreover, in a study conducted in a central business district of downtown Los Angeles (Shoup, 2006), it was shown that in

one year, the cumulative distance drivers travelled while looking for parking spots was equivalent to 38 trips around the world.

Historically, the answer to concerns over traffic congestion caused by a lack of parking spaces has been to increase the parking supply (Parking, 2016). This does not, however, solve the problem. On the one hand, it is true that too few parking spaces can result in increased travel time, but on the other hand, it is now widely recognised that simply providing more parking spaces is not sufficient (or possible) and in some ways can actually contribute to congestion levels.

The parking problem, and its relation to driver satisfaction, air pollution, economic implications, accessibility, and other services of the city, represents a serious challenge in the new domain of *smart cities*. Smart mobility is one of the six areas presently recognised by the European Union as a target for better future urban development around the world (Neirotti, De Marco, Cagliano, Mangano, & Scorrano, 2014).

Much work has been done with respect to the issue of the allocation of parking spots. However, the majority of the work has focused on the planning of parking guidance and information systems in private parking (Abidi, Krichen, Alba, & Bravo, 2017; Abidi, Krichen, Alba, & Molina, 2015; Ni, Sun, & Peng, 2015). Another widely followed line of research is related to the estimation and mapping of parking spots in the city and the status of each one (available or not) (Pham, Tsai, Nguyen, Dow, & Deng, 2015; Zou, Kaffle, Wolfson, & Lin, 2015).

This is a complex problem, as evident in the lack of a practical model whose mathematical definition would allow for the expansion of a line of research. In this paper, similar works are reviewed and a novel optimisation problem linked to real-world application is offered. A bio-inspired algorithm was also developed, which heuristically assigns parking spots available in a city not only from the point of view of users (taking into account their preferences) but also from a global point of view (city policies), as will be seen later. Many tests were carried out under different working conditions to show the practicality of the suggested problem definition and the usefulness of this initial algorithm proposed to solve it.

Literature survey

Solving problems related to parking has become an increasingly important research issue in recent years, as can be seen by analysing the number of publications between 2001 and 2015 on parking issues. It was found that in 2001 only 19 papers on this theme were published, while by 2015 that number had grown to 147. Thus, the number of publications in journals over the last 15 years has increased by more than 700%. This unequivocally demonstrates the expanding interest in providing solutions to the growing problem of parking.

After a thorough study of the literature, we found that most of the research on parking problems can be classified into two broad groups. First, there are studies that concentrate on the identification and mapping of parking spots (Shoup, 2006; Bechini, Marcelloni, & Segatori, 2013; Suhr & Jung, 2014; Pham et al., 2015; Zou et al., 2015). In general, what researchers seek is to know with some degree of accuracy what parking spots are in a certain area in order to help users find a place to park. To create maps or determine where parking is possible, researchers have employed various technologies such as the use of special sensors in the parking spots (IoT) that determine whether a spot is available (Shoup, 2006; Suhr & Jung, 2014; Pham et al., 2015; Zou et al., 2015). Also, popular is the use of video cameras to identify parking spots (Lixia & Dalin, 2012). In Shao, Yang, Zhang, and Ke (2016), a model using private spots during the daytime is proposed, with the premise that the users drive their cars to work.

Second, in the other group, we include papers related to helping users choose an available spot (Fu, Chen, & Sun, 2014a; Fu, Chen, Sun, & Yang, 2014b; Abidi et al., 2015; Ni et al., 2015). In these approaches, researchers have developed proposals to guide users to spots using the information on available parking spots, as well as the requests of the user. However, none of these proposals have considered the simultaneous requests of the users (which would be necessary in any real

system); thus, their proposed solutions (spots assigned to every user) are far from optimal or useful. Shin and Jun (2014) propose an algorithm for parking guidance taking into account several factors like the distance and time of travelling, cost of parking, traffic congestion, etc. In Aydin, Karakose, and Karakose (2017), an Internet-based genetic algorithm (GA) approach is proposed for parking vehicles. In Reis, Sargento, and Tonguz (2018), the cars are used like units to form a self-organised parking network. In Liu, Gao, Wu, and Lin (2018), a model using the sensor data is used to predict the availability of parking spots. In Thomas and Kovoov (2018), a genetic approach is developed to automatically find a parking spot in a parking lot. In Zheng and Geroliminis (2016), an urban parking scenario is modelled with dynamic conditions like traffic jams and variable demand; although this study has interesting results, it is difficult to put into practice. In Mejri, Ayari, Langar, and Saidane (2016), simulated annealing (SA) is used to optimise the assignment of parking spots to users. In Amini and Karabasoglu (2018), the authors developed an integral framework for minimising the cost of charging and transport for electric vehicles. They utilised Dijkstra's algorithm and did a survey of related works (for example, Amini, Moghaddam, & Karabasoglu, 2017).

For another survey of parking solutions, we refer to Lin, Rivano, and Le Mouél (2017) and Velasco and Deniz (2017). Finally, in early 2017, Google announced that it had created a service that determines the levels of availability of parking depending on the geographical area; the vast possibilities of this technology are yet to be understood.

Contribution

The main contribution of the modelling and solving algorithm in the present research is to utilise the information about available spots as well as consider the requests of all users, who are simultaneously looking for an available parking spot in the city. In the next section, the definition of the problem of optimal allocation of parking spots is introduced.

The general idea of the algorithm can be seen in Figure 1. First, in (a), the users request a spot available in the system, via the Internet (b). In (c), the position of the cars with all the available spots is taken as input and sent to the GA. In (d), the GA processes all the data and optimises the general problem. In (e), a solution for every user is sent to them. This paper concentrates on parts (c) through (e).

Paper organisation

The rest of this article is organised as follows: The second section outlines the Optimal Allocation of Parking Spots problem; the third section details the Genetic Algorithm proposed

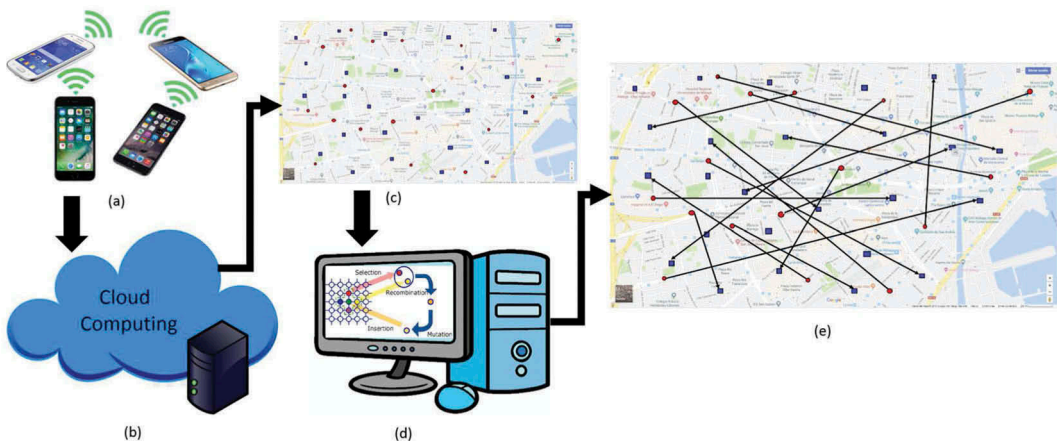


Figure 1. General schema of our approach.

to solve the problem. The fourth section discusses other algorithms applied to this problem; in the fifth section, we show and explain the urban scenario and the instances used in the work presented here. The experimentation methodology used for testing the performance of the GA, and a comparative analysis of the results supported by non-parametric statistical tests, can be found in the sixth section. Finally, the final section sets forth some of the conclusions that can be drawn.

Optimal allocation of public parking spots

In this section, a novel definition of the problem of optimal parking spot allocation is described. First, it is shown how city objects are represented as mathematical objects; then, the problem definition (global optimisation) is presented, as well as an analysis of the size of the search space. Finally, we presented a complexity study of the proposed problem to fully characterise it. Next, we provide a notation of our main variables:

- G Graph representing the streets and corners.
- V Set of vertices representing the corners.
- E Set of edges representing the streets.
- e An edge or street.
- N_e Number of lanes of a street e .
- T_e Traffic load of a street e .
- $D_{traffic}$ Density of traffic of a street e .
- R Reliability of the avenue; it is the probability that all lanes are available.
- v_{max}^e Maximum speed of a street e .
- A_t^e Average time that a vehicle travels a street e .
- κ_e Is the cost of an edge e .
- Δ Matrix of costs.
- C Set of cars $c \in C$.
- S Set of spots $s \in S$.
- $\sigma_{u,v}$ Cost of the minimal path between u and v .

City representation

Putting it simply, the problem of the optimal allocation of parking spots can be understood as follows: (1) there exists a set of drivers and their preferences about a desired parking place; (2) there is a set of public parking spots. The goal is to optimally allocate the best parking spot for each driver, minimising a given cost function. In order to solve the optimal allocation of parking spots problem, it is necessary to have a model of the city and a simulation of this city, in order to design a system amenable for real service.

To carry out the simulation of the allocation of parking spaces and the effect on the dynamics of the city, one should start with the map of a real city. To illustrate this, we use here a map of Malaga, Spain, like that shown in [Figure 2](#). The section of interest of the map for this study comprises the following elements:

- Set of streets
- Direction of the traffic
- Intersections between streets (corners)

In addition to this, the dynamic traffic also involves traffic lights, pedestrians, different kinds of vehicles (cars, motorcycles, buses, etc.) as well as the traffic of the city itself. This is known as



Figure 2. Section of interest of Malaga city.

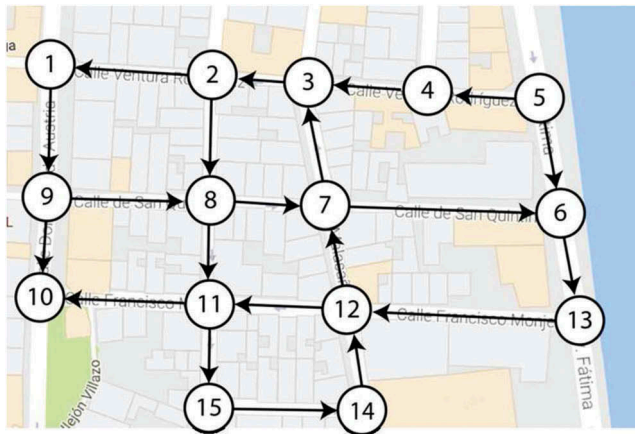


Figure 3. Graph of Figure 2.

a *microanalysis* of the city, as the aim is to have a scientific, yet also a realistic statement of the problem for a near-future implementation of the model.

To model the streets and avenues of the city, a weighted directed graph $G = (V, E)$ was used, where $V \neq \emptyset$ is the set of vertices (corners) and E is a set of edges or streets derived from the streets. The weight assigned to each of the arcs $e = (u, v) \in E$ represents the cost necessary (not in monetary terms but rather in terms of difficulty) to move from an intersection of the city to another, and the direction of the arcs defines the directions of the traffic. Figure 3 shows the graph that models the city in Figure 2.

Weight of the edges

Moving a car from an initial position to a particular parking spot involves a cost that is derived from multiple components such as distance from the original position to the parking spot, traffic blocking the way to the parking spot, number of traffic lights that he/she must negotiate, time of day, day of the week, and area of the city.

One standard way to determine the cost of the edges E of the graph G (see the section on City representation) used to represent the city is to use the distance between two nodes or corners, that

is, the length of the street. However, in this paper, a novel method is proposed that takes into account not only the street's length but also other real aspects like:

- Number of lanes available
- Typical traffic load
- Maximum speed allowed
- Length of the street
- Reliability of the street

The density of traffic $D_{traffic}$ estimated for a street is based on both the number of lanes N_e and the traffic load T_e (see Equation (1)). The T_e is defined as the number of cars circulating per unit of time on the street.

$$D_{traffic} = \frac{T_e}{N_e} \quad (1)$$

The reliability of the avenue $0 < R \leq 1$ is the probability that all lanes are available, meaning that the traffic is not (partially) interrupted by stoppages caused by maintenance, repair, marches, etc. Therefore, as the reliability decreases, the probability of congestion occurring increases (Equation (2)).

$$D_{traffic}(R) = \frac{1}{R} D_{traffic} \quad (2)$$

Theoretically speaking, the average time that a vehicle should take to enter and leave a street is a function of the length l_e of the street e and the maximum speed v_{max}^e at which it is possible to circulate on this street (see Equation (3)).

$$A_t^e = \frac{l_e}{v_{max}^e} \quad (3)$$

On the basis of Equations (2) and (3), Equation (4) is proposed for estimation of the edge's cost.

$$\kappa_e = A_t^e D_{traffic}(R) \quad (4)$$

Thus, the matrix of costs Δ relative to graph G used to model the city is defined as shown in Equation (5).

$$\Delta_{ij} = \begin{cases} 0 & i = j \\ \kappa_e & i \neq j \quad \forall (v_i, v_j) \in E \\ \infty & i \neq j \quad \forall (v_i, v_j) \notin E \end{cases} \quad (5)$$

Calculating the cost of parking a car

Given a weighted graph $G = \{V, E\}$ where $V \subset \mathbb{N}$ and $V \neq \emptyset$ represent the set of crossings of a city and E represents the set of streets connecting such crossings, the cost $\sigma_{u,v} \in \mathbb{R}^+$ of parking a car c in a given spot s was calculated. The car and the spot positions were in $\{u, v\} \in V$. The shortest path $P_{u,v} \subseteq E$ between the position u of the car c and the position v of the parking spot s was calculated by using the algorithm of Dijkstra, and then, the sum of the costs κ_e (see Equation (5)) was found as shown in Equation 6.

$$\sigma_{u,v} = \sum_{e \in P_{u,v}} \kappa_e \quad (6)$$

The optimisation problem

In short, the optimal allocation of a set of parking spots $S = \{s_i \mid i \in \mathbb{N}\}$ for a set of cars $C = \{c_j \mid j \in \mathbb{N}\}$ can be defined as a combinatorial optimisation problem, which aims to find a permutation $\pi \in \Phi_{\Pi}$ of size $|S|$ (see Mahdoun, 2006), where Φ_{Π} is the set of all possible solutions within the search space, i.e., all the potential ways in which cars c_i can be allocated to the available spots s_i , i.e., $\pi = [c_1 \cdots c_{|S|}]$.

The π solution could be expressed as a matrix $W_{|S| \times |C|}$ where $w_{ij} \in \{0, 1\}$, $i = 1, \dots, |S|$, and $j = 1, \dots, |C|$. If $w_{ij} = 1$, this means that spot i is assigned to car j . If $\sum_{j=1}^{|C|} w_{ij} = 0$, then spot i has not been assigned to any car.

Figure 4 illustrates a simple example of an assignment with three cars and four spots. In it, car 1 is assigned to spot 1, car 2 to spot 4, and, finally, car 3 is assigned to spot 2 (spot 3 is empty). In this example, $|S| = 4$, so the solution that represents this mapping will be $\pi = [1, 3, 0, 2]$.

Figure 5 presents the solution $\pi = [1, 3, 0, 2]$ as a matrix W . In W , as mentioned above, each column represents a car $\{1, 2, 3\}$, while each row $\{1, 2, 3, 4\}$ represents a parking spot. As illustrated in this example, for spot 1 and car 1, the value of the cell is 1, which means that car number one is assigned to spot number one and so on for the rest of the cells; in this way, it is possible to transform the permutation π in the matrix W .

Formally, the problem for the optimal allocation of public parking spots (OAPPS) is to find a matrix W that minimises Equation (7), subject to the constraints listed in Equations (8) and (9).

$$\min \left(\sum_{i=1}^{|S|} \sum_{j=1}^{|C|} \sigma_{ij} w_{ij} \right) \quad (7)$$

Subject to:

$$\sum_{i=1}^{|S|} w_{ij} \leq 1 \quad j = 1 \dots |S| \quad (8)$$

$$\sum_{j=1}^{|C|} w_{ij} \leq 1 \quad i = 1 \dots |C| \quad (9)$$

where σ_{ij} is the cost of parking the car j in the spot i , and w_{ij} is 1 if spot i has been assigned to car j (0 otherwise).

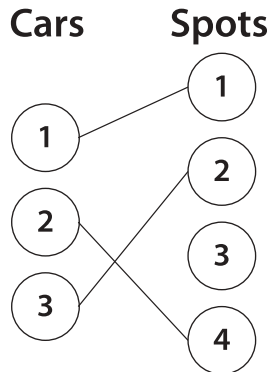


Figure 4. Sample assignment.

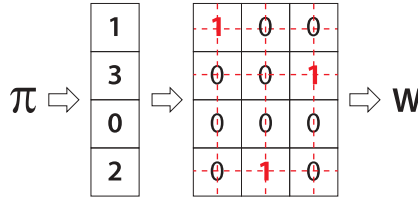


Figure 5. From permutation π to matrix W .

Size of the search space

As shown in Equation (10), to determine the search space size $|\Phi_{II}|$ of the OAPPS problem, three aspects must be studied: (i) if the number of cars is less than the number of spots available, then the size of the search space is $|S|P_{|C|}$, that is, the number of ways of obtaining an ordered subset of $S_{|C|}$ spots of length $|C|$ from a set of S ; (ii) if the number of cars $|C|$ equals the number of available parking spots $|S|$, then the size of the search space is given by the number of permutations possible for the set of S parking spots, that is, the factorial of the number of parking spots, i.e., $|S|!$; finally, (iii) if the number of cars is greater than the number of parking spots, then the size of the search space will be given by $|C|P_{|S|}$.

$$|\Phi_{II}| = \begin{cases} |S|P_{|C|} & |C| < |S| \\ |C|! & |C| = |S| \\ |C|P_{|S|} & |C| > |S| \end{cases} \quad (10)$$

The number of ways $|R|P_{|r|}$ of obtaining a subset of r elements from a set R of size $|R|$ can be calculated as shown in Equation (11).

$${}_n P_r = \frac{n!}{(n-r)!} \quad (11)$$

Figure 6 presents a logarithmic graph which shows an example of the size of the search space for a total of 50 parking spots. The black line shows the growth of the search space for a total of 49 cars, red line for 40 cars, and so on. In all those cases, we observe a light slowdown peak when the number of cars equals the number of spots, which is due to the change in the size of the search space, as shown in Equation (10).

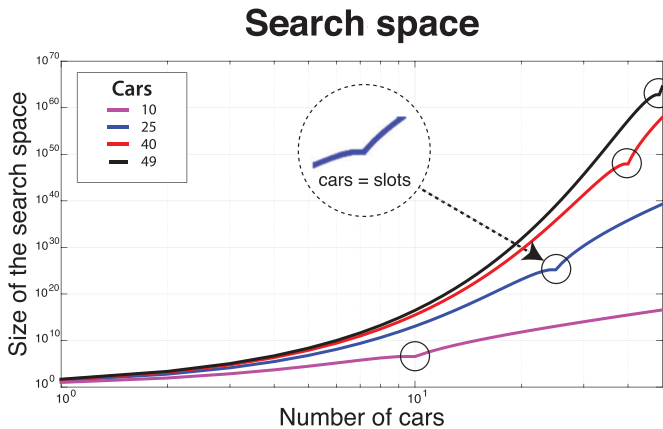


Figure 6. $|\Phi_{II}|$ for 50 spots.

In conclusion, the size of the search space grows depending on the relationship between the number of cars and parking spots: if the number of cars is less than the number of parking spots, the search space grows exponentially; if the number of cars is equal to the number of parking spots, then the growth is factorial; and in any other case, the growth is polynomial. A clear conclusion of this study is that using any brute-force method when the number of spots and cars is large makes the problem intractable by (even intelligent) enumeration.

Proof of NP-completeness

In the previous sections, the problem of the OAPPS has been presented; the current section introduces a proof that the proposed problem is NP-complete. Since this is the first time that this problem has been modelled, it is important to discuss this proof to fully characterise OAPPS.

The process of devising an NP-completeness proof for a problem Q comprises the following steps (Gary & Johnson, 1979):

- showing that Q is in NP ,
- selecting a known NP-hard problem Q' ,
- constructing a polynomial time reduction f from Q' to Q .

Theorem 1. The optimal allocation of public parking spots Q is NP-complete.

Proof: A decision problem is a question in a formal system with a yes-or-no answer, depending on the values of some input parameters, and this is NP if the instances where the answer is ‘yes’ have efficiently verifiable proofs. More precisely, these proofs have to be verifiable by deterministic computations that can be performed in polynomial time. For the problem presented in this paper, the solution π can be verified using the method shown in Equation (12). The temporal complexity of Equation (12) grows linearly with respect to the size of the permutation π that is $O(n)$, so it is clear that the problem is NP .

$$\sum_{i=1}^{|\pi|} \sigma(i, \pi_i) \quad (12)$$

Now that we know that OAPPS is NP, we continue with the proof of NP-completeness. For this, we have selected the NP-complete problem Q' : the *maximum generalised assignment problem* (mGAP), that is well known to be NP-complete (Fisher, Jaikumar, & Van Wassenhove, 1986). It is a generalisation of the assignment problem (Cattrysse & Van Wassenhove, 1992) in which both tasks and agents have a given size. Moreover, the size of each task may vary from one agent to another.

In words, mGAP can be described as follows: we have n kinds of items d and m kinds of containers f . Each container f_i is associated with a capacity k_i . For a container f_i , each item d_j has a profit p_{ij} and a weight λ_{ij} . A feasible solution is one in which, for each container f_i , the total weight of assigned items is at most y_i (its capacity). The goal is to find a maximum profit feasible solution.

Formally, the mGAP problem is to find the matrix X (where $x_{ij} = 1$ if container f_i has item d_j , and 0 otherwise) that maximises the value given by Equation (13) subject to the restrictions of Equations (14) and (15).

$$\sum_{i=1}^m \sum_{j=1}^n p_{ij} x_{ij} \quad (13)$$

$$\sum_{j=1}^n \lambda_{ij} x_{ij} \leq y_i \quad i = 1, \dots, m \quad (14)$$

$$\sum_{i=1}^m x_{ij} = 1 \quad j = 1, \dots, n \quad (15)$$

Now, we can address step 3 since we have mGAP (a known NP-complete problem) as our Q' problem: let us reduce it to our OAPPS problem as follows:

- the set of items $D = \{d_1, \dots, d_n\}$ will be the set of cars $C = \{c_1, \dots, c_n\}$,
- the containers $F = \{f_1, \dots, f_m\}$ will be the parking spots $S = \{s_1, \dots, s_m\}$,
- the capacity of the containers y_i will be 1 since it is only possible to park only one car per spot,
- each car c_j has a cost of parking depending on the spot s_i ,
- the weight of each car λ_{ij} will be 1.
- the total weight of assigned cars to each spot must be less than or equal to one since a single spot can be or empty or occupied, i.e., $y_i \leq 1$.

It is easy to see that these six steps in the reduction occur in polynomial time.

Therefore, there are clear similarities between Equation (13) on mGAP and Equation (7) from OAPPS, as Equation (16) attests:

$$\sum_{i=1}^m \sum_{j=1}^n p_{ij} x_{ij} = \sum_{i=1}^{|S|} \sum_{j=1}^{|C|} \sigma_{ij} w_{ij} \quad (16)$$

Yet each car needs a spot to park, therefore $\lambda_{i,j} = 1 \quad \forall i, j; i = 1, \dots, m, j = 1, \dots, n$. For example, for the image shown in [Figure 4](#), the value of the matrix λ is shown in Equation (17).

$$\lambda = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (17)$$

$$\sum_{j=1}^n \lambda_{ij} x_{ij} = \sum_{j=1}^n \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} x_{ij} \quad (18)$$

Replacing Equation (17) in (14) (see Equation (18)), it is clear that Equation (14) is equal to Equation (8) (see Equation (19)).

$$\sum_{i=1}^{|C|} w_{ij} = \sum_{j=1}^n \lambda_{ij} x_{ij} = \sum_{j=1}^n x_{ij} \quad (19)$$

In conclusion, it can be seen that by applying the technique of restriction described in Gary and Johnson (1979) we can make a reduction from mGAP to the OAPPS, thus concluding that the latter is also an NP-complete problem.

An algorithm for solving the OAPPS problem

We did not want to simply offer a new model for the OAPPS problem, but rather to continue the natural research chain of solving it by designing a GA. For this, we have used Binary Tournament as the selection operator, PMX Crossover as the recombination operator, an exchange of the values of

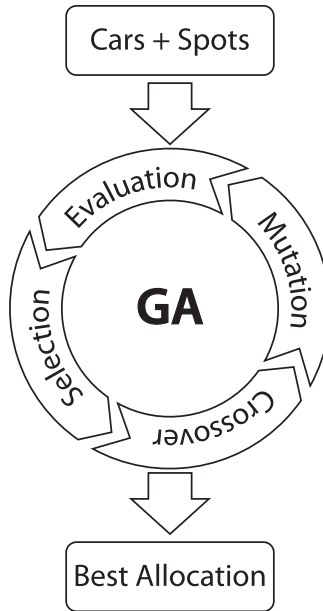


Figure 7. Our approach.

two positions as the mutation operator, as well as an elitist replacement policy. As shown in [Figure 7](#), the algorithm has two inputs. The first is information about the cars: start position, desired target spot (driver request), and car ID (see the section on Encoding to solve the OAPPS problem). The second input is the information about the spots: the spot ID and the node of the city where the spot is located (see the section on Encoding to solve the OAPPS problem). Our GA searches for the best allocation of parking spots and returns the best-found result.

Encoding to solve the OAPPS problem

The public parking spots $S = \{s_1, \dots, s_m\}$ available in the city are stored in a list, as shown in [Figure 8](#). Each item in the list consists of two fields: the parking ID and the identifier of the node in the graph to which each the spot belongs.

The set of cars $C = \{c_1, \dots, c_n\}^T$ are also stored in a list (see [Figure 9](#)). The elements of this list are structured in three fields: the *Car ID*, the identifier of the nearest node, and the identifier of the spot preferred by the user (the target).

The start position for each car *Node ID* is the nearest node in the path (corner). For example, at the moment the driver makes a request ([Figure 10](#)), the car is between the nodes 6 (position u) and 7 (position v); therefore, the nearest node to the car is 6.

Representation of an individual

Each individual in the population (see [Figure 11](#)) is represented by a vector of dimension m , which stores the permutation formed for the set of car IDs. If the number of cars is greater than the number of spots, only the first m elements of the permutation are stored in the individual. A permutation element whose value is zero means there is no car assigned to the spot identified by the index of that component.

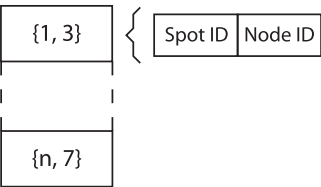


Figure 8. Data structure for storing parking spots.

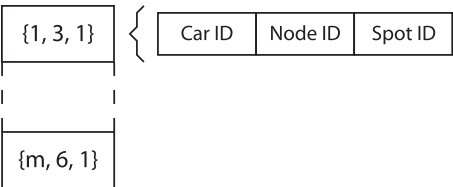


Figure 9. Data structure for storing cars.

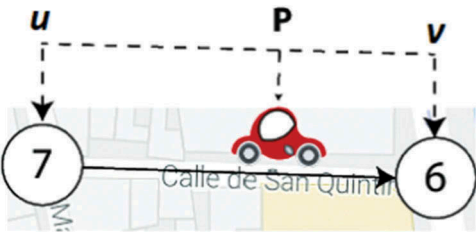


Figure 10. Car starting position is u .

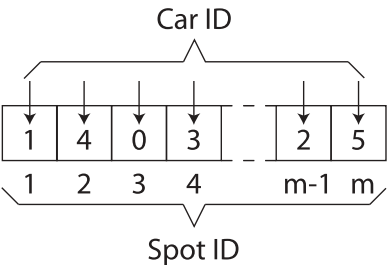


Figure 11. Representation of an individual.

Population

In Figure 12, the population used by the proposed algorithm is represented by a matrix of $n \times |PSize|$ elements, where n again represents the number of parking spots and $PSize$ represents the number of individuals in the population.

Variation operators

As mentioned, individuals of the population are represented by permutations of the car's ID looking for a parking spot. On the one side, for recombination, the algorithm uses the Partially Matched Crossover

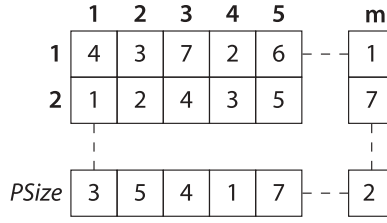


Figure 12. Representation of the population.

(PMX) method (Sivanandam & Deepa, 2007). In PMX, two parents **p1** and **p2** are aligned, and two crossover points **a** and **b** are selected uniformly at random along the length of the parents. The two crossover points give a matching selection, which is used to affect a cross through position by position exchange operations. In-between the crossover points, the genes are exchanged, i.e., 3 and 2, 5 and 6, 1 and 7 exchange places. This is done by mapping parent p2 to parent p1. Now mapping parent p1 to parent p2, 7 and 1, 2 and 3, 6 and 5 exchange places (see Figure 13).

On the other side, in the exchange mutation operator, two random positions of the individual **A** and **B** are chosen and the values corresponding to those positions are interchanged, as shown in Figure 14.

Fitness function

The fitness function is shown in Equation (20) (minimisation problem). For all cars encoded in the individual, we perform the accumulated sum of cost δ for the shortest paths between the node where the car is, at the time of the request, named sp , and the node to which the assigned spot belongs ap , as well as the cost between the node to which the desired spot ds belongs and the assigned spot. For each position ij of the δ matrix, the shortest path between nodes i and j is calculated with Dijkstra's algorithm (Brandes, 2001) and the edge values are calculated by using the equation proposed in 4.

$$F(\pi) = \sum_{i=1}^{|\pi|} \delta(sp_i, ap_i) + \delta(ap_i, dp_i) \quad (20)$$

For example, *car* c_1 wants to park in *spot* s_1 ; however, this spot is not available. According to Figure 15 and in the case that there is just one request for a parking spot, the cost (fitness value) of assigning

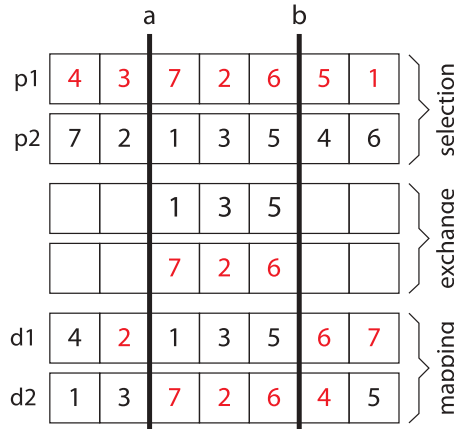


Figure 13. PMX recombination method.

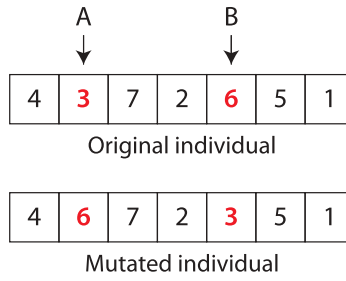


Figure 14. Mutation operation (swap).

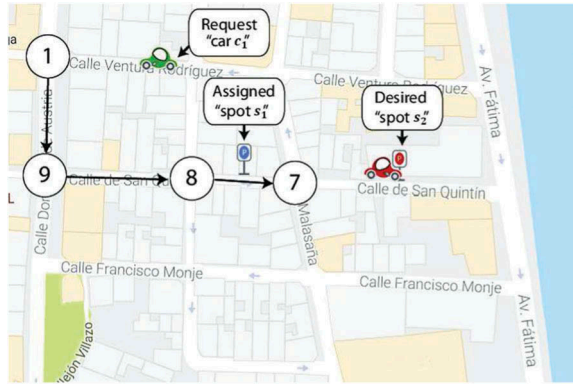


Figure 15. Fitness example.

$spot\ s_1$ to $car\ c_1$ is calculated by adding the cost of parking $car\ c_1$ in the assigned spot, i.e., $\delta(sp, ap)$, plus the cost of moving from the assigned spot to the desired spot, i.e., $\delta(ap, dp)$. In this case, at the time of the request, the nearest node sp in the path of $car\ c_1$ or 'Node ID' (See Figure 9) is node 1, i.e., $sp = 1$. The desired spot dp or 'Spot ID' (see Figure 9) belongs to node 7, i.e., $dp = 7$. Lastly, the assigned spot ap belongs to node 8 ($ap = 8$).

The pseudo-code in algorithm 1 shows the steps followed by the search for the optimal allocation of parking spots.

Algorithm 1 Pseudo-code of our GA

```

1: GenerateInitialPopulation( $P(0)$ )
2: Evaluate( $P(0)$ )
3: while not TerminationCriterion( $P(t)$ ) do
4:    $P^I(t) \leftarrow \text{Select}(P(t))$ 
5:    $P^{II}(t) \leftarrow \text{PMX}(P^I(t))$ 
6:    $P^{III}(t) \leftarrow \text{Exchange}(P^{II}(t))$ 
7:   Evaluate( $P^I(t)$ )
8:    $P(t+1) \leftarrow \text{Replace}(P(t), P^{III}(t))$ 
9:    $t \leftarrow t + 1$ 
10: end while

```

Competing algorithms

To evaluate the effectiveness of the proposed GA's ability to solve the OAPPS problem and perform a sanity check, two alternative methods were compared: (i) Random Search (RS) and (ii) SA. Although this investigation's main focus was the problem characterisation (from theory to smart mobility implementation), and the GA is simply an initial proposal, it was deemed important to compare potential competitors to show that the present work has several added values.

Random Search

RS algorithms are useful for many ill-structured global optimisation problems with continuous and/or discrete variables, but we include it here to perform a *sanity check* on the validity of our algorithm. RS assigns parking spots to cars by generating various random configurations, meanwhile saving the best options, and finally reporting the most optimal solution. We have set a limit for the fitness function evaluations based on the number of the parking spots as shown in Table 3, which is the same number as the GA.

Simulated Annealing

SA is a probabilistic technique for approximating the global optimum of a given function and is often used when the search space is discrete. At each step, the SA solver considers a neighbour's state n' of the current state n and probabilistically decides between moving to state n' or staying in state n . This basic behaviour ultimately leads the system to move to states of lower energy. Typically, this step is repeated until the system reaches a solution that is sufficient for the application or until a given computation budget has been exhausted (Kirkpatrick, Gelatt, & Vecchi, 1983). We have implemented SA to obtain assignments of parking spots. Once again, as for RS, we have set a limit on the fitness function evaluations as a function of the number of parking spots, shown in Table 3, for the search process.

Urban scenario

In order to evaluate these algorithms, an urban scenario was designed which encompasses an area of about 1 km². Twenty-four cases have been created, which are divided into eight groups, each with a different number of cars and parking spots. The reason for dividing the cases into groups is to gradually increase the stress to which the algorithm is subjected, allowing for various situations in the city to be simulated, such as (i) the number of spots available is less than the number of requests, which is the most common situation in a city; (ii) the number of spots is equal to the number of requests; and (iii) the number of spots is greater than the number of requests. These were created randomly in an effort to simulate real-world scenarios.

To create the parking spots, a uniform probability distribution was used: the position of the available parking spots has been defined so that the probability of an available spot is the same in any area of the urban scenario. In the same way, the position of the cars at the time of making a request for the allocation of a spot also has a uniform probability distribution. These configurations have been defined to perform a comprehensive study of the behaviour of the proposed algorithm without it being affected by density produced by a specific probability distribution. The configuration of the instances is shown in Table 1. The data for the initial allocation of cars, preferences, and costs for each of the spots can be viewed and downloaded from the ParkNet website (<http://neo.lcc.uma.es/parknet/desktop>).

Forthcoming experiments appear in the section on Conclusions and future work.

Table 1. Instances.

Instance	Spots	Cars
1	50	30
2		50
3		100
4		50
5		100
6	150	150
7		100
8		150
9		200
10		150
11	200	200
12		250
13		200
14		250
15		300
16	300	250
17		300
18		350
19		300
20		350
21	350	400
22		350
23		400
24		450

Experimental results

Once the problem and the solving algorithms were characterised and defined, the results of a set of experiments were analysed to better understand the nature of OAPPS. This section, therefore, describes the parameters and context used for each of the experiments conducted, as well as the statistical tests applied to validate the results.

Experimental setting

All experiments were performed on a 16 PC cluster with 64 bits Ubuntu Linux v12.10 operating system, each one with an Intel(R) Core(TM)2 Quad processor running at 2.66 GHz with 2 GB of RAM. For the three solvers (GA, RS, and SA), 100 independent runs were performed for each of the 24 instances, amounting to a total of 7200 experiments. The value of each parameter was set to its recommended value (from the literature) as shown in Table 2, the number of the fitness evaluations is shown in Table 3, and all algorithms were programmed with MATLAB.

Statistical analysis of experimental results

In order to compare the experimental results from all the algorithms tested, two statistical analyses were done. The first involved all *classic statistical* indicators (minimum, maximum, mean, median,

Table 2. GA parameters.

Parameter	Value
Population size	100
Crossover probability	0.8
Mutation probability	0.1
Crossover operator	PMX
Mutation operator	Exchanging
Selection method	Binary tournament

Table 3. Number of maximum fitness evaluations.

Spots	Evals
50	3000
100	4000
150	6000
200	9000
250	11,000
300	15,000
350	18,000
400	25,000

Table 4. Average Friedman ranking.

Algorithm	Ranking
GA	14.7917
SA	34.2083
RS	60.5000

Table 5. Adjusted p -values (Holm).

i	Algorithm	Unadjusted p	p_{Holm}
1	RS	0	0
2	SA	0.00131	0.00131

and standard deviation); the second analysis involved *non-parametric indexes* based on hypothesis tests, such as the Aligned Friedman Test and the Holm test.

The table in the Appendix shows a set of classical metrics summarising the 100 independent runs of the algorithms. The first conclusion was that, as expected, the worst performance was that of the RS algorithm since it had the highest cost for all experiments. This confirms that the proposed GA approach has more intelligence in comparison to a random search. The best performance out of all three algorithms was achieved by the GA. Again, the algorithm with the lower standard deviation (higher robustness) was the GA, which is a desirable feature of algorithms aimed at real applications. Based on the data in the Appendix, it can, therefore, be concluded that, for the varied instances considered, the best performing algorithm was the GA.

The second analysis involved the computation of a non-parametric ranking based on hypothesis tests, which can indicate whether there is a significant difference between the performances of the algorithms, as well as identify the best performing solver (see [Tables 4](#) and [5](#)). The Aligned Friedman Test is intended to detect differences between the results obtained by each experiment. Based on the results of these analyses, it was concluded with statistical confidence that GA was the best algorithm.

However, Friedman failed at times to identify specific differences between the best-ranked algorithm (GA) and the others. A Holm analysis was, therefore, used to compare each pair of algorithms. The confidence level for all comparisons was set to 95% ($p_{value} = 0.05$), which ensured that the algorithms were statistically different if their results lied within $p_{value} < 0.05$. [Table 5](#) shows the resulting p_{value} for each set of compared algorithms, with results clearly indicating that p_{value} was lower than 0.05, therefore refuting the (null) hypothesis. Thus, the GA is statistically different and better with respect to SA and RS.

As a final confirmation of the superiority of the GA approach, [Figure 16](#) presents the distribution of fitness for each instance. In [Figure 17](#), the gap of RS and SA is graphed, and again, the best solution is GA. For each of the graphs in [Figure 18](#), one can observe that the GA converges properly before the evaluation budget is attained.

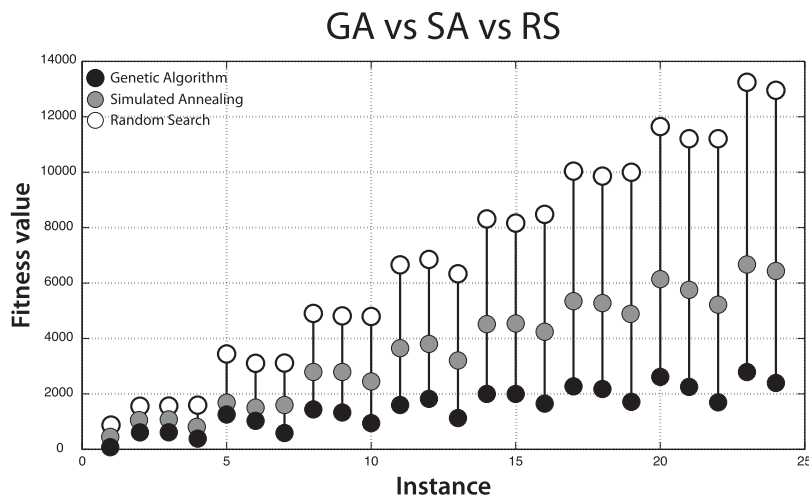


Figure 16. RS vs. SA vs. GA.

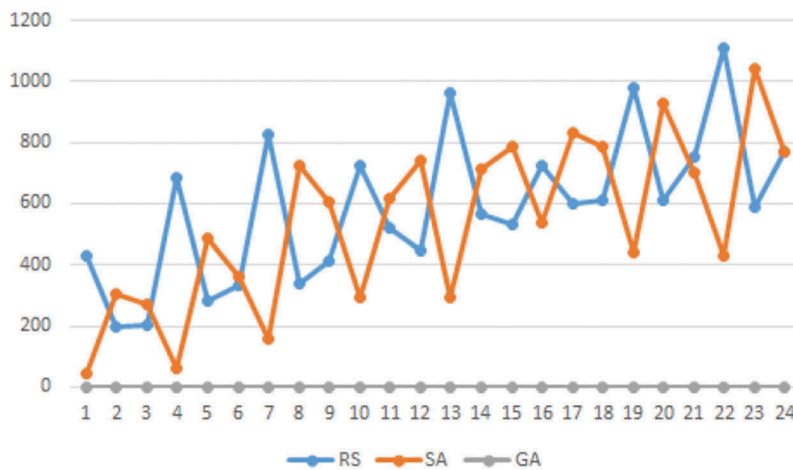


Figure 17. Gap of RS and SA against the best solution of GA.

Conclusions and future work

In conclusion, this research presents a new approach to solving the OAPPS problem: a fascinating and important challenge for modern cities. The intent of this project was to balance scientific theory with the much-needed next step of putting this theory into practice in a real urban location.

Practicality and feasibility of the approach

- Practicality. For one side, the system was designed with the objective to address (and help to solve) a real problem in urban areas. Our results are encouraging: many tests that we have carried under different conditions show that our problem definition is practical and that the algorithms proposed are useful to solve it. On the other side, the new technologies and software, like the simple system of parking designed by Google, indicate an important

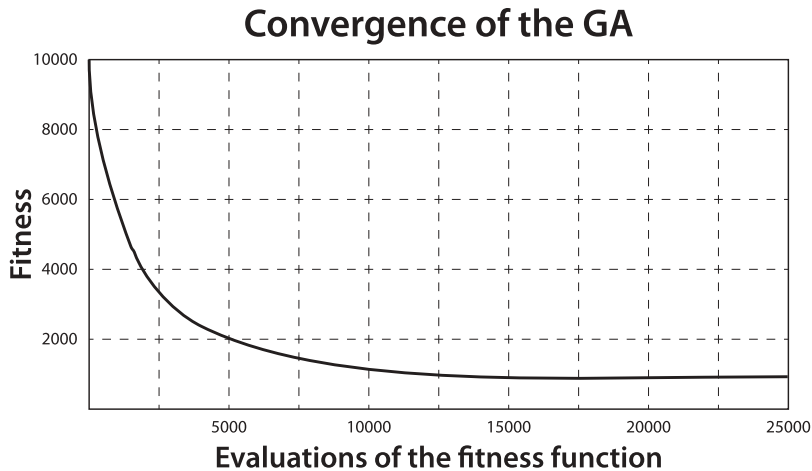


Figure 18. GA for 400 parking spots and 400 cars.

emerging trend in this theme. Our approach can provide a solution that minimises the time of seeks parking lot.

- **Feasibility.** In order to ensure the feasibility of our approach, we modelled real cases of parking availability. Also, there are related papers (see the section on Literature survey) working on the demand in real time, and their results are promising. This shows the plausibility of our approach. Finally, we are encouraged to take the system from an experimental setup to its real use in a mid-size city, and we are considering the necessary additional work:
 - Improve the experimental setting, broaden the algorithms now used, and carry out a more detailed analysis of mobility of cars when searching for a parking spot, as outlined in the section on Future work.
 - As a case study, the map of a city such as downtown Chetumal, Mexico, would be an ideal option for its similarity to the cases used in the present research.
 - Realise that cities differ strongly in their dynamics, urban layout, and availability of parking spots, so that our solution will need to be fitted to each situation.
 - The same city exhibits important changes in its dynamics through time, implying that, for real deployment, our system must use actual measurements throughout the year.

Main contributions

- The OAPPS problem has been introduced, modelled, and characterised (section on Proof of NP-completeness) as an NP-complete problem.
- The OAPPS problem was modelled, generating a set of varied instances, which were solved (section on An algorithm for solving the OAPPS problem).
- The classical statistical analysis (section on Statistical analysis of experimental results) shows that
 - (a) the GA has more intelligence than a random search;
 - (b) the best performance of the tested algorithms belongs to the GA;
 - (c) the algorithm with the lowest standard deviation was the GA.

The conclusion is that the best performing algorithm is the GA.

- The analysis with non-parametric indexes (section on on Statistical analysis of experimental results) shows that
 - (a) according to the Aligned Friedman Test ([Table 4](#)), the GA is the best algorithm, with statistical confidence;
 - (b) a Holm analysis ([Table 5](#)) shows that the GA is different and better than SA y RS.

- With respect to fitness, [Figure 18](#) shows that the GA converges properly before the evaluation budget is attained.
- The solution proposed (GA) is simple and efficient. It was compared (using the same overall solution and experimentation benchmark, section on Urban scenario) with RS y GA. After an initial experimentation phase, over 24 instances, three algorithms, and three search conditions (more than 7000 experiments), we obtained a body of results (section on Conclusions and future work) that statistically proves that the GA is an adequate proposal for the problem.

Future work

- Change in the experimental setting (section on Experimental setting). Different types of probability distributions will be used to carry out a study of the algorithm under specific conditions and responding to specific events.
- A more realistic map of the city is already being developed.
- SUMO is also being utilised (Krajzewicz, Erdmann, Behrisch, & Bieker, 2012) to make a highly detailed analysis of cars and mobility while the drivers are looking for their parking spots.
- The set of algorithms used to solve the problem should be broadened to see if GA is still competitive against them.
- Finally, more scenarios (than the 24 studied here) should be investigated since all cities are highly dynamic, and open data from actual measurements throughout the year should be considered.

Acknowledgements

The first author wishes to express his gratitude to Colegio de la Frontera Sur ECOSUR for its support, as well as Casey Gibson for her invaluable technical support.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work has been partially funded by the Spanish MINECO and FEDER project [TIN2014-57341-R] (<http://moveon.lcc.uma.es>); MICINN CI-RTI national network on smart cities [TIN2016-81766-REDT] (www.cirti.es).

ORCID

Javier Arellano-Verdejo  <http://orcid.org/0000-0002-4920-283X>

Federico Alonso-Pecina  <http://orcid.org/0000-0002-3402-2959>

References

- Abidi, S., Krichen, S., Alba, E., & Bravo, J. M. M. (2017). A hybrid heuristic for solving a parking slot assignment problem for groups of drivers. *International Journal of Intelligent Transportation Systems Research*, 15(2), 85–97.
- Abidi, S., Krichen, S., Alba, E., & Molina, J. M. (2015). A new heuristic for solving the parking assignment problem. *Procedia Computer Science*, 60, 312–321.
- Amini, M. H., & Karabasoglu, O. (2018). Optimal operation of interdependent power systems and electrified transportation networks. *Energies*, 11(1), 196.
- Amini, M. H., Moghaddam, M. P., & Karabasoglu, O. (2017). Simultaneous allocation of electric vehicles parking lots and distributed renewable resources in smart power distribution networks. *Sustainable Cities and Society*, 28, 332–342.
- Aydin, I., Karakose, M., & Karakose, E. (2017, April). A navigation and reservation based smart parking platform using genetic optimization for smart cities. In *Smart Grid and Cities Congress and Fair (ICSG), 2017 5th International Istanbul* (pp. 120–124). IEEE, Istanbul, Turkey.

- Bechini, A., Marcelloni, F., & Segatori, A. (2013, October). *A mobile application leveraging QR-codes to support efficient urban parking*. In Sustainable Internet and ICT for Sustainability (SustainIT) (pp. 1–3). IEEE, Palermo, Italy.
- Brandes, U. (2001). A faster algorithm for betweenness centrality*. *Journal of Mathematical Sociology*, 25(2), 163–177.
- Cattrysse, D. G., & Van Wassenhove, L. N. (1992). A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60(3), 260–272.
- Falcochchio, J. C., & Levinson, H. S. (2015). *Road traffic congestion: A concise guide* (Vol. 7). Springer, Germany.
- Fisher, M. L., Jaikumar, R., & Van Wassenhove, L. N. (1986). A multiplier adjustment method for the generalized assignment problem. *Management Science*, 32(9), 1095–1103.
- Fu, J., Chen, Z., & Sun, R. (2014a, October). *Research on intelligent terminal oriented optimal parking space recommendation model*. In Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on (pp. 2373–2378). IEEE, Qingdao, China.
- Fu, J., Chen, Z., Sun, R., & Yang, B. (2014b). Reservation based optimal parking lot recommendation model in Internet of vehicle environment. *China Communications*, 11(10), 38–48.
- Gary, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman & Co. New York, NY, USA
- Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680.
- Krajewicz, D., Erdmann, J., Behrisch, M., & Bieker, L. (2012). Recent development and applications of SUMO-Simulation of Urban MObility. *International Journal on Advances in Systems and Measurements*, 5(3–4), 128–138.
- Künzli, N., Kaiser, R., Medina, S., Studnicka, M., Chanel, O., Filliger, P., ... Schneider, J. (2000). Public-health impact of outdoor and traffic-related air pollution: A European assessment. *The Lancet*, 356(9232), 795–801.
- Lin, T., Rivano, H., & Le Mouél, F. (2017). A survey of smart parking solutions. *IEEE Transactions on Intelligent Transportation Systems*, 18(12), 3229–3253.
- Liu, K. S., Gao, J., Wu, X., & Lin, S. (2018, June). On-street parking guidance with real-time sensing data for smart cities. In 2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON). IEEE, Hong Kong, China.
- Lixia, W., & Dalin, J. (2012, November). *A method of parking space detection based on image segmentation and LBP*. In Multimedia Information Networking and Security (MINES), 2012 Fourth International Conference on (pp. 229–232). IEEE, Nanjing, Jiangsu, China.
- Mahdoun, A. (2006). *FRANZ ROTHLAUF representations for genetic and evolutionary algorithms* (pp. Hardbound). Springer Verlag, Germany. ISBN 3-540-25059-X 325.
- Mejri, N., Ayari, M., Langar, R., & Saidane, L. (2016, September). Reservation-based multi-objective smart parking approach for smart cities. In *Smart Cities Conference (ISC2), 2016 IEEE International* (pp. 1–6). IEEE, Trento, Italy.
- Neirotti, P., De Marco, A., Cagliano, A. C., Mangano, G., & Scorrano, F. (2014). Current trends in smart city initiatives: Some stylised facts. *Cities*, 38, 25–36.
- Ni, X. Y., Sun, D. J., & Peng, Z. R. (2015). An improved incremental assignment model for parking variable message sign location problem. *Journal of Advanced Transportation*, 49(7), 817–828.
- Parking. (2016). International parking institute — Serving professionals in parking [online]. Retrieved from <http://www.parking.org/>
- Pham, T. N., Tsai, M. F., Nguyen, D. B., Dow, C. R., & Deng, D. J. (2015). A cloud-based smart-parking system based on Internet-of-Things technologies. *IEEE Access*, 3, 1581–1591.
- Reis, A. B., Sargento, S., & Tonguz, O. K. (2018). Smarter cities with parked cars as roadside units. *IEEE Transactions on Intelligent Transportation Systems*, 19(7), 2338–2352.
- Shao, C., Yang, H., Zhang, Y., & Ke, J. (2016). A simple reservation and allocation model of shared parking lots. *Transportation Research Part C: Emerging Technologies*, 71, 303–312.
- Shin, J. H., & Jun, H. B. (2014). A study on smart parking guidance algorithm. *Transportation Research Part C: Emerging Technologies*, 44, 299–317.
- Shoup, D. C. (2006). Cruising for parking. *Transport Policy*, 13(6), 479–486.
- Sivanandam, S. N., & Deepa, S. N. (2007). *Introduction to genetic algorithms*. Springer Science & Business Media, Germany.
- Suhr, J. K., & Jung, H. G. (2014). Sensor fusion-based vacant parking slot detection and tracking. *IEEE Transactions on Intelligent Transportation Systems*, 15(1), 21–36.
- Thomas, D., & Kooor, B. C. (2018). A genetic algorithm approach to autonomous smart vehicle parking system. *Procedia Computer Science*, 125, 68–76.
- Velasco, F., & Deniz, O. (2017, June 14–16). Existing approaches to smart parking: An overview. In *Smart Cities: Second International Conference, Smart-CT 2017. Proceedings* (Vol. 10268, p. 63). Mlaga, Spain: Springer.
- Zheng, N., & Geroliminis, N. (2016). Modeling and optimization of multimodal urban networks with limited parking and dynamic pricing. *Transportation Research Part B: Methodological*, 83, 36–58.
- Zou, B., Kafle, N., Wolfson, O., & Lin, J. J. (2015). A mechanism design based approach to solving parking slot assignment in the information era. *Transportation Research Part B: Methodological*, 81, 631–653.

Appendix

Statistical Results

Table A1. Classical statistical measures.

Spots	Cars	Std. measure	RS	SA	GA
50	30	Min	779.70	189.80	147.30
		Max	921.00	547.90	1001.10
		Mean	877.14	344.80	286.10
		Median	883.10	338.90	278.30
		Std	23.40	62.44	110.78
	50	Min	1431.90	588.90	485.82
		Max	1612.40	925.80	1713.80
		Mean	1555.46	761.71	680.37
		Median	1562.30	745.60	653.32
		Std	32.21	73.47	177.20
	100	Min	1488.20	553.50	492.15
		Max	1630.40	1004.00	1737.70
		Mean	1562.58	778.17	696.16
		Median	1565.45	774.45	653.65
		Std	30.80	88.04	183.66
100	50	Min	1529.30	428.60	194.36
		Max	1654.60	772.20	1905.40
		Mean	1595.69	625.32	458.32
		Median	1597.40	627.90	416.86
		Std	21.64	75.90	227.05
	100	Min	3362.10	1378.20	878.92
		Max	3504.80	1955.50	3841.90
		Mean	3440.87	1685.46	1337.23
		Median	3446.30	1683.80	1257.92
		Std	31.56	107.92	436.20
	150	Min	2979.90	1194.60	688.85
		Max	3159.60	1776.70	3566.80
		Mean	3104.44	1482.20	1105.89
		Median	3110.20	1474.90	1028.85
		Std	36.75	112.32	376.62
150	100	Min	3029.50	1288.70	327.54
		Max	3168.80	1870.10	3536.60
		Mean	3114.49	1602.75	832.86
		Median	3115.95	1614.55	819.04
		Std	28.68	111.60	385.30
	150	Min	4813.70	2453.40	1099.80
		Max	4973.00	3053.50	5409.00
		Mean	4903.71	2783.29	1743.15
		Median	4904.55	2793.95	1657.80
		Std	34.16	123.54	561.07
	200	Min	4677.20	2487.30	917.23
		Max	4888.30	3217.20	5384.30
		Mean	4816.90	2789.60	1612.82
		Median	4823.70	2786.60	1495.23
		Std	40.85	132.34	600.24
200	150	Min	4698.40	2144.90	570.33
		Max	4859.50	2972.80	5308.50
		Mean	4793.91	2449.29	1338.72
		Median	4798.40	2437.50	1269.33
		Std	30.74	141.03	652.39
	200	Min	6490.00	3194.50	1047.10
		Max	6742.30	3934.60	7245.50
		Mean	6657.17	3650.45	2051.58
		Median	6657.50	3651.20	1912.60
		Std	47.06	157.10	893.03
	250	Min	6706.50	3302.20	1227.50
		Max	6927.20	4418.80	7545.60
		Mean	6852.48	3804.18	2195.89
		Median	6858.70	3818.95	2067.00

(Continued)

Table A1. (Continued).

Spots	Cars	Std. measure	RS	SA	GA
250	200	Std	46.49	178.79	817.15
		Min	6207.30	2850.50	584.49
		Max	6425.10	3567.50	6945.10
		Mean	6335.71	3205.42	1532.82
		Median	6342.50	3194.95	1412.99
	250	Std	45.85	153.03	817.56
		Min	8128.20	4028.50	1224.60
		Max	8411.60	5062.20	8983.70
		Mean	8315.10	4541.31	2418.49
		Median	8320.25	4534.00	2191.60
	300	Std	48.05	194.53	1074.55
		Min	8015.50	4162.90	1271.00
		Max	8248.90	4934.40	8877.50
		Mean	8163.20	4518.02	2503.02
		Median	8165.40	4503.70	2417.50
300	250	Std	47.44	162.25	947.10
		Min	8285.30	3895.10	1005.50
		Max	8567.80	4732.50	9159.00
		Mean	8476.68	4241.89	2285.13
		Median	8487.65	4237.90	2078.00
	300	Std	55.50	174.17	1213.65
		Min	9882.90	4978.80	1414.30
		Max	10,120.00	6015.60	10,763.00
		Mean	10,037.70	5346.04	2807.43
		Median	10,047.50	5326.40	2475.30
	350	Std	50.85	192.11	1384.59
		Min	9671.40	4812.40	1359.10
		Max	9956.40	5652.60	10,649.00
		Mean	9860.20	5274.64	2830.39
		Median	9872.35	5280.70	2525.10
350	300	Std	60.37	198.75	1373.06
		Min	9842.50	4337.80	910.09
		Max	10,100.00	5351.30	10,748.00
		Mean	9996.13	4890.19	2484.57
		Median	10,002.50	4888.60	2373.59
	350	Std	54.22	209.82	1199.67
		Min	11,479.00	5657.70	1614.60
		Max	11,772.00	6568.10	12,444.00
		Mean	11,648.26	6138.77	3337.73
		Median	11,651.50	6139.60	3256.60
	400	Std	54.23	200.49	1374.51
		Min	11,087.00	5302.90	1297.50
		Max	11,297.00	6234.30	12,036.00
		Mean	11,207.69	5754.61	2967.04
		Median	11,211.00	5757.85	2762.50
400	350	Std	45.92	204.93	1351.99
		Min	11,083.00	4767.70	918.67
		Max	11,304.00	5599.20	12,000.00
		Mean	11,206.70	5219.73	2728.54
		Median	11,211.50	5206.30	2488.67
	400	Std	41.09	187.98	1449.20
		Min	13,086.00	6110.00	1907.00
		Max	13,332.00	7306.10	14,095.00
		Mean	13,244.83	6668.73	3819.54
		Median	13,251.00	6678.35	3612.50
	450	Std	48.04	191.47	1644.10
		Min	12,788.00	5954.70	1466.20
		Max	13,051.00	6963.60	13,890.00
		Mean	12,956.46	6434.63	3384.80
		Median	12,959.50	6405.00	2982.70
		Std	53.90	233.56	1621.08