Omega ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Omega

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

Parking slot assignment for urban distribution: Models and formulations ☆

Mireia Roca-Riu [a,b,*], Elena Fernández [b], Miquel Estrada [a,c]

[a] Center for Innovation in Transport (CENIT), C/Jordi Girona 1-3, C3 S120, 08034 Barcelona, Spain
[b] Barcelona Tech-UPC, Department of Statistics and Operations Research, C/Jordi Girona 1-3, C5, 08034 Barcelona, Spain
[c] Barcelona Tech-UPC, Department of Transport and Regional Planning, C/Jordi Girona 1-3, B1, 08034 Barcelona, Spain

ARTICLE INFO

Article history:
Received 7 October 2013
Accepted 21 April 2015

Keywords:
Parking slot assignment problem
Scheduling
Time windows
Mathematical programming formulations

ABSTRACT

A key element to enhance urban distribution is the adequate management of parking space, particularly for loading and unloading operations. An in-advance booking system able to be adjusted to users needs can be a very useful tool for city councils. Such a tool should be fed with criteria for allocating requests to time slots. In this paper we discuss alternative criteria for the parking slot assignment problem for urban distribution and we propose the use of mathematical programming formulations to model them. Several models are proposed, analyzed and compared among them. Extensive computational experience is presented with a detailed analysis and comparison, which provides quantitative indicators of the quality of each of the proposed models.

1. Introduction

Major cities face multiple problems caused by delivery operations in urban distribution. In many cases city councils regulate the conditions under which carriers may operate, and then each carrier acts as his own decision maker, scheduling his operations according to established rules and his own resources' limitations. In contrast, city councils must act as joint decision makers for all operations that must be carried out with the use of public space, since a criterion on how to allocate public space among carriers is needed. The adequate management of public space is crucial for successful urban distribution. The lack of parking facilities has been pointed out among the aspects with higher impact in urban delivery (see, for instance, [1]). This paper deals with the problem of allocating public parking space in the streets during the loading and unloading hours for goods distribution from an operations research perspective. The use of operations research in urban planning has been discussed in [2] in contrast to interactive city planning, reacting to events as they occur [3].

A general framework for the problem that we address is the following. The goal of the city council is to regulate the use of public space in order to prevent carriers from parking illegally and to improve urban distribution. To this end a system is proposed in which carriers can only park at designated parking areas and designated time periods, which are assigned in advance by the city council. Such a system would eliminate the very negative effects in traffic flow due to carriers double parking and would also benefit carriers greatly, since available parking space would be guaranteed at designated time periods. For this, some public space, consisting of a set of parking places, is allocated for loading and unloading operations at a given area during some hours each day. The city council asks carriers to express in advance their requests for a parking time interval and to inform about the duration of their operations. These durations will take into account not only loading and unloading activities but also movement times between the parking space and the operation site. Then, each carrier is assigned a time interval, based on his preference. Since carriers will know in advance their assigned time interval, they will be able to re-optimize their routes beforehand so as to arrive on time to the assigned parking space. Thus, we assume that carriers will accept and respect the assigned intervals, even if they do not fit their requests (Fig. 1).

Several works point out the advantages of allocating specific parking facilities for carriers in order to reduce the negative impact of distribution operations. As an example, the results of [4] illustrate the positive response of most drivers and truck operators to one such initiative in Kobe (Japan) in 2001. It is however clear that, even if a set of parking places is allocated,

☆This manuscript was processed by Associate Editor Salazar-Gonzalez.

* Corresponding author at: Barcelona Tech-UPC, Department of Statistics and Operations Research, C/Jordi Girona 1-3, C5, 08034 Barcelona, Spain.
E-mail addresses: mireia.roca-riu@upc.edu (M. Roca-Riu), e.fernandez@upc.edu (E. Fernández), miquel.estrada@upc.edu (M. Estrada).

http://dx.doi.org/10.1016/j.omega.2015.04.010
0305-0483/

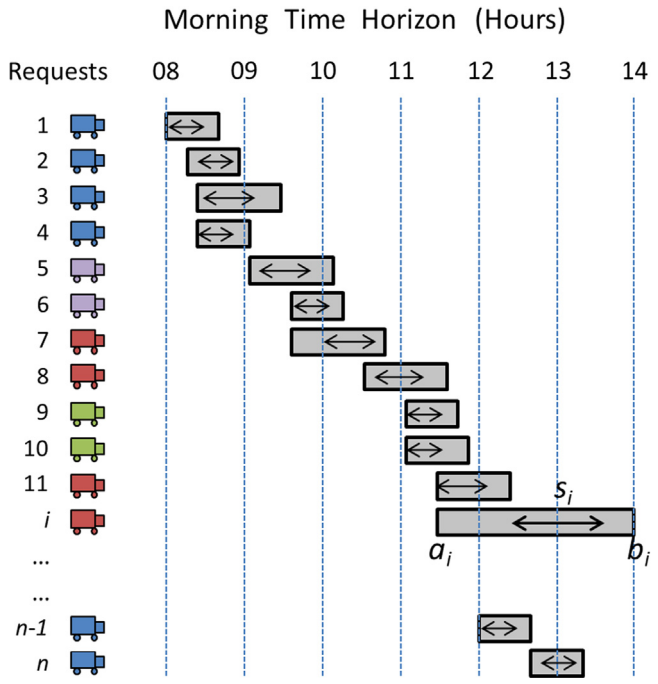

Fig. 1. Requests of different companies along morning time horizon.

negative effects will prevail when carriers arrive to delivery areas but find no available parking space. The benefit of establishing some booking system for allocating in advance places has been assessed in several practical studies. The results of a pilot test carried out in 2005 in Dos Hermanas, Sevilla (Spain) [5] confirm the effectiveness of the internet booking system implemented for the assignment of parking space in load areas of the city center. An advanced booking system was theoretically studied in [6] for the city of Winchester (UK) in the Highstreet area. Supported by an EU project [7], a recent eight months pilot test in Bilbao (Spain) successfully trialled a booking system for carriers in four zones of the city. Free areas could be assigned to users without pre-booking, and pre-booked carriers could be reassigned if they were out of schedule, as well.

In the above referenced works priority lists have been used for deciding the assignment of places to requests. However, alternative criteria or techniques can be applied for establishing the allocation of parking space to carriers. In this paper we propose the use of mathematical programming optimization models for solving the parking slot assignment problem. To the best of our knowledge this problem has not been addressed so far in the literature in the context of urban distribution. Nevertheless, we can find some similarities with other problems studied in the literature. For instance, the problem that we address can be seen as a particular case of a scheduling problem with time windows (see, for instance, [8]). Further, the concept of earliness/tardiness, as it has been used in scheduling problems with time windows [9,10] or other contributions in flow shop scheduling problems based on the manufacturing industry [11,12] can also be exploited in our case as we will see later on.

Problems like the Berth Allocation Problem (BAP) [13], the Aircraft-Gate Allocation Problem (AGAP) [14], or the allocation of trains to platforms at rail stations (Train Platforming Problem [15]) also have some similarities with the Parking Slot Assignment Problem (PAP). In all these problems it is assumed that the arrival times of the vehicles as well as the durations of their operations are known in advance, and the vehicles have to be assigned to some facility for a given time. The BAP aims to optimally schedule and assign vessels to berthing areas along a quay. The most common objective in the BAP is to minimize total service time. This objective favors the assignment of higher priorities to vessels with smaller handling volumes than to vessels with larger handling volumes [16]. Because this type of solution may not satisfy the ocean carrier' preferences, another studied objective is the minimization of the deviation from the preferred berth [17]. Some other works consider, in addition, objective functions with penalties for unsatisfied time windows [18]. Similar characteristics are present in the AGAP [14], in which the gates where aircrafts will stop are planned taking into account different criteria: efficiency of flight schedules, passenger walking distance, or robust use of the gates in front of disruptions [19]. The distinctive feature of the PAP with respect to the above problems is that the carriers' time windows are flexible, in the sense that the parking times assigned to the carriers by the city council may not coincide with the requested ones. Still we assume that carriers accept and respect the assigned intervals, provided these are known in advance. The reason for this assumption is that carriers can adapt their routes in advance so as to arrive on time to the assigned parking space. This assumption does not hold in the BAP, the AGAP or the allocation of trains to platforms, whose time windows are not flexible and thus must be respected when making the assignment. In the BAP, while advancing the arrival date to port is usually not feasible, postponing it typically implies very high costs. The same happens with the departure dates from the port due to contractual agreements between port operators and ocean carriers. Something similar happens with aircrafts, where indeed flight schedules are not planned according to the availability of gates at the airports.

The main question that we address in this work is the criterion that should be optimized. Given that the goal of the city council is to avoid illegal parking, and carriers would greatly benefit from having a prebooked parking space, any solution satisfying all requests within their time windows would be optimal. From this point of view, one would think that the problem we face reduces to a feasibility problem. However, a given instance may not have an assignment satisfying all requests. What should the outcome be in this case? What can the decision maker do if there is no parking slot for everyone? Some fair criterion is needed in order to allocate carriers' requests when their needs cannot be satisfied. Several modeling alternatives are proposed and compared in this work, in which we restrict to one single loading and unloading area with several parking places. Each of these alternatives aims at optimizing a different criterion, which may seem appropriate for finding compromise solutions for the parking allocation problem. First, we propose a mathematical programming formulation for the feasibility problem under consideration and introduce the alternative models based on different optimization criteria. Then, we study some relations among the different models and give a sufficient condition for unfeasibility of the basic model. Afterwards, we carry out a thorough comparison, based on the results of an extensive computational experimentation, which provides quantitative indicators of the quality of each of the proposed models.

In this paper we assume that data is deterministic. This is indeed a simplifying modeling assumption as the nature of the problem involves some uncertainty, particularly with respect to vehicle arrival times. However, as we will see, the deterministic formulations proposed are already complex and difficult to solve.

The remainder of this paper is structured as follows: Section 2 formally introduces the problem and presents a mathematical programming formulation for the feasibility problem as a mixed integer problem (MIP). Section 3 presents the different alternative models that have been considered, and studies some of their properties. Section 4 describes the computational experiments we have run, and presents the obtained results together with an extensive analysis and comparison, and Section 5 presents a simple heuristic and compares its solutions with those obtained previously. We close the paper in Section 6 with some comments and possible avenues for further research.

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

2. Definition and formulation of the Parking Slot Assignment Problem

The Parking Slot Assignment Problem (PAP) is defined as follows. We consider one loading and unloading (L/U) area with c common parking places that can be used by carriers for their L/U operations. Let $[0,T]$ denote the time interval when L/U operations must be scheduled. Let also Q, with $|Q|=q$, denote the index set of L/U operations, each of them with a request for parking assignment within time period $[0,T]$. Associated with each request $i \in Q$, the parameters $[a_i, b_i]$ and s_i, respectively, denote the time window for the beginning of the operation i and its duration (Fig. 1). Since b_i is the latest instant when the beginning of operation i can be scheduled, operation i can last until b_i+s_i. Feasible solutions to the PAP consist of assignments of requests to parking places within the time period $[0,T]$, that satisfy the time window for the beginning of each request and such that at each time slot no more than c parking places are occupied.

We formulate the problem of finding a feasible assignment as a vehicle routing problem with time windows [20] in which customers represent the carriers' requests and each vehicle represents a parking place. Thus, each *route* is equivalent to a sequence of carriers' requests which are served consecutively at the same parking space. To formulate the problem we define an auxiliary complete directed network $N=(V,A)$, with set of vertices $V=Q\cup\{v_d\}$, where v_d plays the role of the depot for these fictitious routes, but has no real meaning. The arcs of A can be classified in the following types: (a) (v_d,i) with $i\in Q$; (b) (i,j) with $i,j\in Q$; and, (c) (i,v_d) with $i\in Q$. We define two sets of decision variables. Binary variables x_{ij}, for all $(i,j)\in A$, indicate whether or not arc (i,j) is used. The meaning of these variables is the following. When both $i,j\in Q$, $x_{ij}=1$ if and only if the requests of carriers i and j are assigned consecutively to the same parking place. When $i=v_d$, $x_{v_dj}=1$ indicates that the request of carrier $j\in Q$ is the first one in the sequence assigned to some parking space. Finally, when $j=v_d$, $x_{iv_d}=1$ indicates that the request of carrier $i\in Q$ is the last one in the sequence assigned to some parking place. We also define a second set of continuous decision variables t_i, with $i\in Q$, to indicate the starting time for the parking request of carrier $i\in Q$ in the schedule. Then a formulation for the PAP is as follows:

$$\text{minimize} \quad z(x,t) \tag{1}$$

$$\text{subject to} \quad \sum_{j\in Q} x_{v_dj} \le c \tag{2}$$

$$\sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} = 0, \quad i\in Q \tag{3}$$

$$\sum_{(i,j)\in A} x_{ij} = 1, \quad i\in Q \tag{4}$$

$$t_i+s_i-t_j \le (1-x_{ij})M, \quad i,j\in Q,\ (i,j)\in A \tag{5}$$

$$a_i \le t_i \le b_i, \quad i\in Q \tag{6}$$

$$0 \le t_i \le T, \quad i\in Q \tag{7}$$

$$x_{ij}\in\{0,1\}, \quad (i,j)\in A. \tag{8}$$

The objective function $z(x,t)$ will be discussed in the following section depending on the criteria used. In all cases, when the domain defined by constraints (2)–(8) contains some feasible solution the optimal value will be zero. The meaning of the constraints is as follows. Constraints (2)–(4) define the *routes*. In particular, Constraint (2) limits the maximum number of parking places to c, by restricting the number of routes starting at the depot. Constraints (3) guarantee the flow balance along the hypothetical routes. Constraints (4) ensure that all requests are allocated to a parking place, by imposing that exactly one route serves each customer. Constraints (5) relate time variables t with flow variables x, to guarantee that time values respect the sequence of service. M is a big enough value that makes the constraint redundant when $x_{ij}=0$. Note that Constraints (5) also prevent subtours. The time window constraint of each request is imposed in (6). Finally, (7) and (8) define the domain for the t and x variables. Note that if data parameters are integer values, and x are binary, then t are also integer without the need of explicitly imposing it. The reader is addressed to [20] for further details of this type of models. The formulation has $(q+1)q$ binary variables and q continuous variables. The number of constraints is $4q+q(q-1)+1$.

Because it is known that constraints with big M values produce weak linear programming (LP) relaxation bounds, it is convenient to find tight estimations of M. For instance, we can use $M=\max_i b_i+\max_i s_i-\min_j a_j$, (in Section 4.1 alternative values are proposed).

In principle, the LP relaxation of the above formulation could be reinforced with the classical subtour elimination constraints (SEC):

$$\sum_{\substack{(i,j)\in A\\ i,j\in W}} x_{ij} \le |W|-1, \quad W\subset V/\{v_d\}. \tag{9}$$

However, as shown by the Example A.1 in the Appendix, for usual values of the data and the parameter M, the addition of the SECs (9) does not reinforce the LP relaxation of constraints (2)–(8), since the resulting domain will always contain some feasible solution, even if the original domain with the integrality constraints is not feasible. This means that the LP value of the different models we will consider will always be zero, independent of whether or not constraints (9) are used. For this reason in the following Constraints (9) are omitted.

In formulation (2)–(8) the time when each of the requests is allocated to some parking place, t_i, is explicit, while the specific parking place to which it is allocated is not explicit. Note, however, that the set of requests allocated to each parking space can be easily identified by tracing the set of requests of each of the routes. This allocation could have been made explicit in the formulation by defining decision variables x_{ij}^k with $(i,j)\in A$, $k\in\{1,\ldots,c\}$ indicating whether or not request i and j are allocated to parking place k, and request j is scheduled immediately after request i, at the expenses of a considerably larger number of binary variables.

3. Alternative models for the PAP

In this section we propose alternative models to address the PAP. Broadly speaking they differ from each other in the optimization criterion that is considered, which, in turn, may require additional modeling changes. As mentioned before, suitable and fair criteria are not easy to decide. On the one hand, considering the PAP as a feasibility problem would not distinguish among feasible solutions when there are several. Further, such a feasibility problem will not yield any solution for all the cases where no assignment satisfying all requests exists. There is a public resource that should be allocated fairly to several operators. In these cases we will allow solutions with *non-accomplished requests*, i.e. requests that are not scheduled within their requested time windows. This naturally leads to alternatives aiming to reduce the degree of non-accomplishment of requests in a fair way. The concept of fairness has been addressed in optimization, associated with various types of problems, particularly when resources have to be allocated [21]. In our case, we incorporate fairness by resorting to objective functions that *penalize* unfeasible solutions in alternative ways, by using different criteria to quantify their degree of non-accomplishment of requests. For instance, we can

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

minimize the overall non-accomplishment by somehow weighting the *earliness* or *tardiness* of the requests assignments relative to their respective time windows, measured in time units. Alternatively, the objective may focus on the number of non-accomplished requests. When considering these alternative objectives, we will extend the domain for feasible solutions allowing for the violation of time window Constraints (6) and, possibly, introducing additional particular constraints. Often the value of the big constant M has to be recomputed since the proposed value above is no longer valid when Constraints (6) are relaxed. Instead, we can take $M = T + \max_i s_i$, as an upper bound on the maximum allowed time for L/U operations. The optimization models that we propose are described below:

*MOD*0: *feasibility PAP*. Here we just look for a feasible solution to formulation (2)–(8), ignoring any other aspect. Any constant objective function $z^0(x,t) = \kappa$ is appropriate, so throughout we use $\kappa = 0$.

In the following models, Constraints (6) are omitted whereas all other constraints, (2)–(5), (7) and (8), are maintained.

*MOD*1: *earliness/tardiness minimization*. In this model we minimize an objective in which time windows' violations are penalized in proportion to their earliness or tardiness. To this end we introduce one new set of decision variables e_i, $i \in Q$, that represent the earliness/tardiness in the assignment of each request. For $i \in Q$, e_i is defined as the time deviation $a_i - t_i$ between the lower limit a_i and the actual scheduling time of request $i \in Q$, if it is scheduled before a_i, or as the deviation $t_i - b_i$ between the actual scheduling time of request $i \in Q$ and its upper limit b_i if it is scheduled after b_i. That is, e_i can be determined as the maximum of the three values: 0, $a_i - t_i$ and $t_i - b_i$. This expression is not linear, although it can be easily linearized by inclusion of the additional sets of Constraints

$$e_i \geq a_i - t_i, \quad i \in Q \tag{10}$$

$$e_i \geq t_i - b_i, \quad i \in Q \tag{11}$$

$$e_i \geq 0, \quad i \in Q. \tag{12}$$

The objective in MOD1 is the minimization of the overall non-accomplishment, which is a weighted sum of the earliness/tardiness (13). That is

$$z^1(x,t) = \sum_{i \in Q} w_i e_i = \sum_{i \in Q} w_i \max\{0, a_i - t_i, t_i - b_i\} \tag{13}$$

where w_i is the weight associated with request $i \in Q$.

*MOD*2: *minimization of maximum earliness/tardiness*. This model focuses again on a measure of the earliness/tardiness of solutions. Now, instead of considering the overall non-accomplishment, we focus on the maximum non-accomplishment, measured as the maximum earliness or tardiness from the requested time windows, among all requests. This objective can be expressed as:

$$z^2(x,t) = \max_{i \in Q} \max\{0, a_i - t_i, t_i - b_i\}. \tag{14}$$

MOD2 is a bottleneck min–max optimization problem. We minimize the objective function $z^2(x,t)$, defined as the maximum deviation from its requested time window among all carriers. In its turn, for each carrier $i \in Q$, the value of its deviation is $\max\{0, a_i - t_i, t_i - b_i\}$. This inner max only guarantees that the deviation from its preferred time window is computed correctly. Note that when a request is computed inside its requested time window, both $a_i - t_i < 0$ and $t_i - b_i < 0$. In this case, however, the correct value of the deviation is 0.

The objective $z^2(x,t)$ can be useful if small deviations from the requested time windows are not considered important, and the relevant measure of the quality of a solution is the maximum earliness/tardiness among all requests.

As before, this objective function is not linear, but can be easily linearized by extending the set of variables and constraints of MOD1 with one additional variable and one additional set of constraints. Let m denote the maximum non-accomplishment. Thus, the objective in MOD2 is the minimization of m. That is

$$z^2(x,t) = m. \tag{15}$$

Variable m must be related to the remaining variables in the formulation. This can be done, for instance, by means of constraints:

$$m \geq e_i, \quad i \in Q. \tag{16}$$

It is possible to simplify the above formulation by removing all the e_i variables, and using directly m in Constraints (10)–(12). With this we reduce the q continuous variables e_i, and the q Constraints (16). We call MOD2b to the resulting model, where the specific set of constraints is

$$m \geq a_i - t_i, \quad i \in Q \tag{17}$$

$$m \geq t_i - b_i, \quad i \in Q \tag{18}$$

$$m \geq 0. \tag{19}$$

*MOD*3: *earliness/tardiness minimization subject to maximum displacement*. This model tries to somehow address jointly the two main concerns of MOD1 and MOD2, by optimizing the same objective function as in MOD1, the overall non-accomplishment, but limiting the earliness or tardiness in the assignment of any request from its requested time window to a maximum value fixed in advance, d. That is, in MOD3 the objective is:

$$z^3(x,t) = \sum_{i \in Q} w_i e_i = \sum_{i \in Q} w_i \max(0, a_i - t_i, t_i - b_i). \tag{20}$$

Now, in addition to Constraints (10)–(12), which establish the values of variables e_i, $i \in Q$, we include one new constraint for each request limiting its maximum possible earliness or tardiness:

$$e_i \leq d, \quad i \in Q. \tag{21}$$

Note that in this formulation, the value of M can be set to $M = \max_i b_i + 2d + \max_i s_i - \min_j a_j$, because of the new constraint.

*MOD*4: *minimization of number of requests scheduled outside the time window*. Models MOD1, MOD2, and MOD3 quantify the time deviations of the solutions from the requests time windows, but ignore if this non-accomplished demand affects to a small or a large number of requests. In MOD4 we focus on the number of requests scheduled outside the asked time window, rather than on the magnitude of the non-accomplishment. To compute this value, associated with each request $i \in Q$ we define a binary decision variable β_i indicating whether or not request i is scheduled outside its time window. Now the objective that we consider is

$$z^4(x,t) = \sum_{i \in Q} \beta_i. \tag{22}$$

In order to activate the new indicator variables we include the set of constraints:

$$K\beta_i \geq e_i, \quad i \in Q \tag{23}$$

where K is a parameter that must be bigger than e_i for all $i \in Q$. For instance, we can set $K = \max_{i \in Q} \max\{a_i, T - b_i\}$.

Note that like in MOD2 the e_i variables can be removed from Constraints (10)–(12) and use the β_i variables instead. By doing so q continuous variables e_i and the q Constraints (23) are eliminated. We will call the resulting model MOD4b. The constraints that activate the β_i variables are

$$K\beta_i \geq a_i - t_i, \quad i \in Q \tag{24}$$

$$K\beta_i \geq t_i - b_i, \quad i \in Q. \tag{25}$$

An extension of MOD4 and MOD4b arises when each non-accomplished request is weighted by the duration of its associated

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

operation. Then, the objective is

$$z^{4'}(x,t)=\sum_{i\in Q}s_i\beta_i. \tag{26}$$

MOD5: Cost minimization. Except for MOD0, all previous models allow solutions where some request is not scheduled within its time window. An alternative to such relaxed models would be to impose time window Constraints (6), and to outsource additional parking space at the time periods when the city council parking space is insufficient to satisfy the carriers demand. In a realistic scenario, this additional space could be obtained from a nearby public parking or from parking areas for non-commercial private use, next to the city council parking places. The outsourced additional parking places would be reserved for L/U operations for only some hours of the total time horizon. This is the idea of MOD5 in which the objective is to minimize outsourcing costs:

$$\sum_{t=0}^{T}\mu_t n_t. \tag{27}$$

In (27) n_t represents the excess (relative to c) in the number of requests that are scheduled at time slot t and μ_t is the unit cost for each outsourced parking place at time slot $t\in[0,T]$. Therefore, (27) represents the cost for making available additional parking places during some hours of the day so as to eliminate non-accomplishment. Below we present an extension of formulation (2)–(8) suitable for MOD5.

Consider a new set of decision variables y that will be used to represent the *routes* associated with outsourced requests. That is $y_{ij}=1$ if request $j\in Q$ is scheduled immediately after request $i\in Q$ in some outsourced parking place. Then the extended formulation is

$$\text{minimize}\quad z^5(x,y,t) \tag{28}$$

$$\text{subject to}\quad \sum_{j\in Q}x_{v_dj}\le c \tag{2}$$

$$\sum_{(i,j)\in A}x_{ij}-\sum_{(j,i)\in A}x_{ji}=0,\quad i\in Q \tag{3}$$

$$t_i+s_i-t_j\le(1-x_{ij})M,\quad (i,j)\in A \text{ with } i,j\in Q \tag{5}$$

$$a_i\le t_i\le b_i,\quad i\in Q \tag{6}$$

$$0\le t_i\le T,\quad i\in Q \tag{7}$$

$$\sum_{(i,j)\in A}y_{ij}-\sum_{(j,i)\in A}y_{ji}=0,\quad i\in Q \tag{29}$$

$$\sum_{(i,j)\in A}(x_{ij}+y_{ij})=1,\quad i\in Q \tag{30}$$

$$t_i+s_i-t_j\le(1-y_{ij})M,\quad (i,j)\in A \text{ with } i,j\in Q \tag{31}$$

$$x_{ij},y_{ij}\in\{0,1\},\quad (i,j)\in A. \tag{32}$$

As in the previous formulations, Constraint (2) ensures that the number of requests that are not outsourced does not exceed the number of reserved places, c. In addition we have two sets of flow constraints: the previous set (3) for the routes defining the allocation within the reserved parking places, and another set (29) for the outsourced requests. Constraints (30) guarantee that all requests are scheduled, either to a reserved parking place or to an outsourced place. The assigned time slots are now regulated by means of the same sets of Constraints (5) and (6) and the new set (31).

Note that with the new set of decision variables, for $i\in Q$ the sum $\sum_{(i,j)\in A}y_{ij}$, takes the value 0 or 1, and indicates whether or not the request of customer $i\in Q$ is outsourced. If we assume that $\mu_t=1$ for all $t\in\{0,\dots,T-1\}$, i.e. the unit cost for outsourced parking space is the same for all time periods, then

$$z^5(x,y,t)=\sum_{t=0}^{T}\mu_t n_t=\sum_{t=0}^{T}n_t=\sum_{i\in Q}\sum_{(i,j)\in A}y_{ij} \tag{33}$$

which coincides with the objective of MOD4. That is, objective (33) plus Constraints (2), (3), (5)–(7), (29)–(32) is also an alternative formulation for MOD4.

However, the above formulation, as it stands, does not allow us to compute the objective function value $z^5(x,y,t)=\sum_{t=0}^{T}\mu_t n_t$ for the general case when the outsourcing costs μ_t may vary among time periods. In this case we need to know the exact number of requests that are occupying an outsourced parking place at a given time period t. For this we can define an additional set of binary variables O_{it}, $i\in Q$, $t\in\{0,\dots,T-1\}$ to identify the requests that are occupying an outsourced place at a given time period. These new variables are related to the original y variables by means of the constraints

$$O_{jt'}\ge O_{it}-(1-y_{ij}),\quad i,j\in Q,t,t'\in\{0,\dots,T-1\} \text{ with } t'=t+s_i \tag{34}$$

$$O_{it}\le O_{it'},\quad t\in\{0,\dots,T-1\},\quad t+1\le t'\le t+s_i. \tag{35}$$

While Constraints (34) activate the outsourcing indicator variables at the time period when a request is outsourced, Constraints (35) guarantee that the indicator variable of an outsourced request is activated during all the time interval in which the operation associated with this request takes place. Now we can express the objective function as

$$z^{5'}(x,y,t)=\sum_{t=0}^{T}\mu_t\sum_{i\in Q}O_{it}. \tag{36}$$

This formulation has qT new binary variables, and $q(q-1)T+qT\sum_i s_i$ constraints, resulting in many more variables and constraints than the previous ones. Also, the objective function weights should be defined.

In the Appendix, several tables summarize the details of the presented models and objectives. Table 6 presents the parameters and the sets used and Table 7 summarizes the objective functions. Tables 8 and 9, respectively, present all the variables and constraints, grouped by the formulation they belong to. Finally, Table 10 counts the number of variables and constraints of each formulation.

3.1. Relationship among models

The models proposed in Section 3 focus on solving the same problem under different criteria. Therefore, they present some relationships that will be analyzed in this section through the comparison of the domains of their respective formulations. We use Ω^i to denote the feasible domain of MODi. In particular,

$\Omega^0=\{(x,t)$ satisfying (2)–(8)$\}$

$\Omega^1=\{(x,t,e)$ satisfying (2)–(5), (7), (8), (10)–(12)$\}$

$\Omega^2=\{(x,t,e,m)$ satisfying (2)–(5), (7), (8), (10)–(12), (16)$\}$

$\Omega^3=\{(x,t,e)$ satisfying (2)–(5), (7), (8), (10)–(12), (21)$\}$

$\Omega^4=\{(x,t,e,\beta)$ satisfying (2)–(5), (7), (8), (10)–(12), (24), (25)$\}$

$\Omega^5=\{(x,t,y)$ satisfying (2), (3), (5)–(7), (29)–(32)$\}$

Observe that, in general, the above domains are defined in different spaces so they cannot be compared. The exception being Ω^1 and Ω^3, for which we have $\Omega^3\subset\Omega^1$. In order to compare the above domains we consider their respective projections onto the

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

Ω^1 space being

$$\Omega^1_{(x,t)} = \Omega^1 \bigcap \{e_i = 0 \ i \in Q\}$$
$$\Omega^2_{(x,t)} = \Omega^2 \bigcap \{m = 0, \ e_i = 0 \ i \in Q\}$$
$$\Omega^3_{(x,t)} = \Omega^3 \bigcap \{e_i = 0 \ i \in Q\}$$
$$\Omega^4_{(x,t)} = \Omega^4 \bigcap \{e_i = 0, \ \beta_i = 0 \ i \in Q\}$$
$$\Omega^5_{(x,t)} = \Omega^5 \bigcap \{y_{ij} = 0 \ (i,j) \in A \ i \in Q\}$$

Note that when $e_i = 0$ for $i \in Q$, Constraints (10)–(12) reduce to (6) and $\Omega^0 = \Omega^1_{(x,t)}$. With a similar reasoning we can relate Ω^0 to the other domains defined above.

Proposition 1. *For the restricted domains the following relationships hold*:

(1) $\Omega^0 = \Omega^1_{(x,t)}$, (3) $\Omega^0 = \Omega^2_{(x,t)}$, (5) $\Omega^0 = \Omega^3_{(x,t)}$,
(2) $\Omega^0 = \Omega^4_{(x,t)}$, (4) $\Omega^0 = \Omega^5_{(x,t)}$.

Moreover, given a solution $(x,t) \in \Omega^0$, then the extended solution $(x,t,0) \in \Omega^1$, and similar extended solutions can be built for the rest of the domains. Thus, we have

Corollary 1. *If* $\Omega^0 \neq \varnothing$ *then,* $\Omega^i \neq \varnothing$ *for* $i = 1, \ldots, 5$.

As a consequence of the previous proposition, we can obtain further relations. For instance, suppose $\Omega^1 \neq \varnothing$ and its optimal value is 0. Let $(x,t,e) \in \Omega^1$ be an optimal solution. Then $e = 0$, and (x,t) is a solution in Ω^0. With a similar reasoning, we have

Corollary 2. *If* $\Omega^i \neq \varnothing$ *for some* $i = 1, \ldots, 5$ *and its optimum value is* 0, *then*

(1) $\Omega^0 \neq \varnothing$, *and*
(2) $\Omega^j \neq \varnothing$ *and its objective value is* 0 *for* $j = 1, \ldots, 5 \ j \neq i$.

3.2. An unfeasibility condition for MOD0

Below we present a sufficient unfeasibility condition for MOD0. As we have discussed, MOD0 is basically a feasibility model. However, as we will see in the computational experiments section, detecting its unfeasibility is not always easy, even if large computing times are allowed. For this reason an effective condition can be very useful for detecting unfeasible instances.

The main idea is to analyze the requests that must be completed within different time intervals and to compare the time needed to satisfy these requests and the overall time offered in this interval, which depends on the available places. Indeed, if the total time needed for the requests is greater than the time offered, the instance is unfeasible. Before presenting the sufficient condition we introduce some additional notation. Let $Q^{\alpha,\beta} \subset Q$ denote the subset of requests whose time window is contained in the interval $[\alpha,\beta]$. That is, $Q^{\alpha,\beta} = \{i \in Q | [a_i, b_i] \subseteq [\alpha,\beta]\}$.

A lower bound on the time requested within a given interval $[\alpha,\beta]$ is the sum of the durations of the requests that must be completed inside the given time interval, i.e. $L^{\alpha,\beta} = \sum_{i \in Q^{\alpha,\beta}} s_i$. This bound may not be tight, since there may be additional requests for the time interval $[\alpha,\beta]$, when the time window of some request not in $Q^{\alpha,\beta}$ overlaps it, i.e. $[\alpha,\beta] \cap [a_i, b_i] \neq \varnothing$ for some $i \notin Q^{\alpha,\beta}$.

On the other hand, an upper bound of the time that is offered in the interval $[\alpha,\beta]$ is $U^{\alpha,\beta}$, computed as the sum of the duration of the time interval multiplied by the number of places offered, $c(\beta-\alpha)$, plus some extra time. For the extra time we take into account that even if some requests are assigned at the very end of the time interval (time instant β), the assignment will still be feasible since time windows limit the beginning of the service. Indeed, the operations corresponding to such assignments will totally take place after time interval $[\alpha,\beta]$. Thus, to compute the extra time we assume that exactly c requests are assigned just at the end of the interval (one at each place), and that these assignments correspond to the c largest durations of the requests indexed in set $Q^{\alpha,\beta}$. That is, the extra time coincides with the c-centrum [22–25] of the durations of the requests indexed in $Q^{\alpha,\beta}$, that we denote by $c\text{-}s^{\alpha,\beta}$. Hence, if $s_{i_1} \geq s_{i_2} \geq \cdots \geq s_{i_{|Q^{\alpha,\beta}|}}$ are the sorted duration values of the requests indexed in $Q^{\alpha,\beta}$, then $c\text{-}s^{\alpha,\beta} = \sum_{r=1}^{c} s_{i_r}$. Therefore, our upper bound on the time that is offered in the interval $[\alpha,\beta]$ can be expressed as $U^{\alpha,\beta} = c(\beta-\alpha) + c\text{-}s^{\alpha,\beta}$. Our unfeasibility sufficient condition is then as follows:

Proposition 2. *Let* $[\alpha,\beta] \subseteq [0,T]$. *If* $L^{\alpha,\beta} > U^{\alpha,\beta}$ *then MOD0 is unfeasible*.

4. Computational experiments for model comparison

In this section we present the results we have obtained in a series of computational experiments we have run to analyze and compare the different models studied in this paper. First, data generation is described in Section 4.1 and then results are presented in the three subsequent sections. Since the primary objective is to solve the feasibility problem, in Section 4.2 we show the results obtained after one hour of execution of MOD0, plus the evaluation of the unfeasibility condition of Section 3.2. Furthermore, we analyze the effectiveness of the models MOD1–MOD4 within the same maximum computing time. In Section 4.3, we cross-evaluate the solutions given by each of the models MOD1–MOD4. That is, for each instance and model, the best solutions obtained by the other models in the same instance are evaluated. Finally, a sensitivity analysis of the maximum displacement is carried out with models MOD1 and MOD3, which share the overall non-accomplishment criteria in Section 4.4. Models have been implemented in the Optimization Programming Language `OPL` and solved with the commercial software `CPLEX 12.1`. In all the experiments the weight coefficients in the objective functions of MOD1 and MOD3 have been set at value one. All experiments have been run on a PC with two four-core processors (8 threads) running at 2.0 GHz and 16 GB of RAM. Note that the time limit of one hour used would also be realistic in practice, when one solution should be obtained by the city council each working day.

4.1. Data generation

Since we are not aware of any benchmark instances that could be used in our experiments, we generated a set of 60 test instances which follow the patterns observed in an experimental study in the city of Barcelona (Spain) [26]. All instances have a similar structure with parameters randomly generated. Each instance represents an area during a whole day [0,T]. The number of parking places for each instance is uniformly drawn from [2,8], and we assume requests are only made within a subinterval $[\widehat{a},\widehat{b}] \subseteq [0,T]$ corresponding to morning hours from 8 h to 14 h or afternoon hours from 16 h to 20 h. We use minutes as time unit. This provides enough precision and it is operative in practice. Thus, $[0,T] = [0,1440]$ and $[\widehat{a},\widehat{b}] = [480,840]$ for the morning or $[\widehat{a},\widehat{b}] = [960,1200]$ if we also consider the afternoon period.

Then, the total number of requests of each instance is computed. Following the patterns observed in reality [26], requests are distributed according to a triangular pattern around a peak hour that is located either in the center or at the beginning of the morning or afternoon interval, depending on the goods type. At

peak hour, demand is usually higher than the number of available places. Given these distributions and usage levels, a unique parameter defining the overall demand density determines the total number of requests. The overall demand density γ is the number of requests per hour-place, and for each instance it is uniformly drawn from [1.5, 3.5]. For each instance, given the interval $[\hat{a}, \hat{b}]$, γ, and the number of places c, the total number of requests is computed as $|Q| = \gamma(\hat{b} - \hat{a})c$.

Finally, the features for the requests of each instance are generated: duration of the operation, and requested time window for the beginning of the service. Following the study of [26], the duration of an operation is drawn from a normal distribution N (18,5). We assume that the duration of the operation includes all the time needed for the operator to perform the delivery, including the time to cover the distance from the parking place to the customer, or the specific features of the shipments that might involve more or less time. The process for generating the time windows is more complex. As mentioned earlier [26] shows that requests are distributed according to a triangular pattern around peak hours, where the peak hour is located either at the center or at the beginning of the morning or afternoon intervals. Thus, instances have been divided into three types, depending on the distribution of requests along time intervals: (a) *triangular centered*, (b) *triangular asymmetric*, and (c) *double peak*. In the first group, only the morning interval is considered, and the peak hour is located at the middle of the interval (11 h). The second group also considers the morning interval but the peak hour is skewed earlier in the morning (9h30). Finally, double peak considers morning and afternoon subintervals and peak hours are centered in the middle of the respective intervals (11 h and 18 h). Once the type of demand has been set for an instance, for each of its requests we set the midpoint for its time window as well as its width. The midpoint is drawn according to the distribution of its type of demand. Finally, the width of the time windows is set to 20, 40, 60, and 80 min, with probability 0.2, 0.2, 0.5, and 0.1, respectively. Time windows are built symmetrically, centered at the middle of the interval, depending on the time window width. Table 1 summarizes the characteristics of the instances. Note that the overall demand density γ and the duration of the required requests entail different levels of saturation around peak hour. Parameter γ is related to the unfeasibility of the instance, since the higher the density of demand, the more difficult it becomes to give service to all requests.

Next, we discuss the different values of M tested for each of the models. For MOD0, four different values of parameter M for Constraint (5) were preliminarily considered: (a) the unique value for each instance proposed in Section 3, $M = \max_i b_i + \max_i s_i - \min_j a_j$. Adapted values for each subset of constraints: (b) $M_i = b_i + s_i - \hat{a}$ and (c) $M_{ij} = b_i + s_i - a_j$. And, a more general value for all instances based on generation instance parameters: (d) $\overline{M} = \hat{b} + \max_i s_i - \hat{a}$. No significant differences were observed neither in the results nor in the computational times and the more general value $\overline{M}$ was used in all the experiments presented in this section.

Four different values of M were also tested for MOD3. A more general value for all instances: (a) $\overline{M} = \hat{b} - \hat{a} + 2d + \max_i s_i$; two constraint related ones: (b) $M_i = b_i + 2d + s_i - \hat{a}$ and (c) $M_{ij} = b_i + 2d + s_i - a_j$; and the unique value proposed for each instance in Section 3: (d) $M = \max_i b_i + 2d + \max_i s_i - \min_j a_j$. Finally, given that only small differences were obtained with the different values, the general value $\overline{M}$ was chosen.

In the case of MOD1, MOD2 and MOD4 no tighter values for M are possible and the value used was $M = T + \max_i s_i$. As for the parameter K of MOD4, we experimented with value $K_i = \max(a_i, T - b_i)$ but no significant differences were observed and the general value ($K = \max_i \max\{a_i, T - b_i\}$) was used.

4.2. Numerical results

As we will see, the outcome of MOD0 is closely related to the outcome of MOD1–MOD4, so we start this section by analyzing the effect of some of the instances' parameters in the results of MOD0, and the effectiveness of the unfeasibility check of Proposition 2. Fig. 2 relates the status of MOD0 at termination to the value of the demand density parameter γ and the type of requests distribution (triangular centered, triangular asymmetric and double peak). For this, benchmark instances are partitioned into three sets: (a) *Feasible* (top); (b) *Time Limit* (middle), when time limit was reached without knowing whether or not the instance is feasible; and (c) *Unfeasible* (bottom), detected either by CPLEX or by the sufficient condition of Proposition 2. Several intervals $[\alpha, \beta]$ were used in the unfeasibility condition check. The center of all intervals coincides with the peak hour, whereas the interval widths range from a minimum of 40 min to a maximum of 180 min, with checks every 20 min. The computational burden of these tests is negligible as it never exceeds 0.01 s. Slight differences can be observed in Fig. 2 among the three types of benchmark instances (triangular centered, triangular asymmetric and double peak). In the first two groups, there are more instances solved to optimality than in the double peak ones, where more instances reached the maximum time limit.

As it was expected, Fig. 2 shows that instances become more difficult with the increase of demand density. At the extreme values of the parameter γ instances are either optimally solved when $\gamma \in [1.5, 1.75]$, or unfeasibility is proven when $\gamma \in [3.07, 3.5]$. Moreover, none of the instances with $\gamma \in [2.38, 3.07]$ were found to

Table 1
Instances characteristics.

General data				
#	Places (c)	Requests interval $[\hat{a}, \hat{b}]$	Demand density (γ)	Distribution Pattern
20		[480,840]		Centered (11 h)
20	$U[2, 8]$	[480,840]	$U[1.5, 3.5]$	Skewed (9h30)
20		$[480, 840] \cup [960, 1200]$		Centered (11 h,18 h)

Requests data	
Duration of operation (s_i)	TW width
$N[18, 5]$	20% of 20 min 20% of 40 min 50% of 60 min 10% of 80 min

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

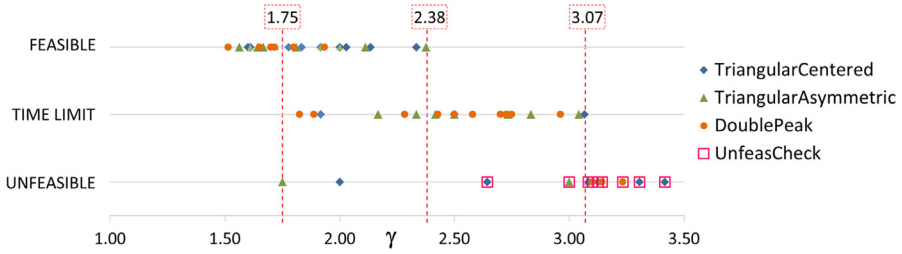

Fig. 2. Results from MOD0.

Table 2
Results of MOD1–MOD4 in the set of instances.

Status	MOD0		Status	MOD1		MOD2		MOD3		MOD4	
	No. Instances	CPU		No.	CPU	No.	CPU	No.	CPU	No.	CPU
			OPT	28	1,028	28	522	28	709	23	425
Feasible	28	108	*TL/S*	–		–		–		5	3,600
			TL/NS	–		–		–		–	
			OPT	–		–		–		–	
Unfeasible	11	3,512	*TL/S*	11	3,600	11	3,600	6	3,600	11	3,600
			TL/NS	–		–		5	3,600	–	
			OPT	1	839	1	98	1	459	1	2,010
Time limit	21	3,600	*TL/S*	20	3,600	20	3,600	17	3,600	20	3,600
			TL/NS	–		–		3	3,600	–	

be feasible. Observe the effectiveness of the unfeasibility check, which was able to detect the unfeasibility of nine instances for which CPLEX 12.1 terminated within the time limit without detecting neither feasibility nor unfeasibility.

Next we analyze the numerical results obtained with CPLEX for models MOD0–MOD4. The results obtained for each model and benchmark instance are presented in Table 11 in the Appendix. The meaning of the first five columns is the following. In the first column (*Id*), each instance is identified with a numerical label. Column *Rq* gives the number of requests, *c* the number of parking places, *Hop* the length of the time horizon, i.e. the number of operating hours, and γ the density.

Column *Type* under MOD0 indicates the status of MOD0 at termination: *F* when a provable optimal solution was found, *U* when unfeasibility was proven by CPLEX, *UC* when unfeasibility was proven by the check of Proposition 2, and *TL* when the time limit was reached but the instance could not be classified in any of the former groups. For each model MODj, $j=1,\ldots,4$, column *SOL* gives the value of the optimal solution, when the value is bold, or the value of the best solution found when the time limit was reached, otherwise. If no solution was found within the allowed computing time the entry in column *SOL* is empty. Negative entries in column *SOL* of MOD0 indicate that the unfeasibility of the instance was proven. This entry is -1 when unfeasibility was detected by CPLEX within the CPU time limit (instances 15 and 30), or -2 when unfeasibility was detected with the test of Proposition 2. For each model, column *TIME* gives the CPU time to termination in seconds or 3600 when time limit was reached.

Table 11 does not include information on lower bounds at termination because, for all models MOD0–MOD4, these bounds were always zero in all the cases that optimality could not be proven.

As can be seen, MOD0 found a provable optimal solution for 28 instances and proved that the instance was not feasible in two cases. The average CPU time for the instances that terminated with a certificate of optimality or unfeasibility is 148.1 s. However, for the remaining 30 instances it terminated without knowing whether or not the instance was feasible, even if the feasibility check indicates that 9 such instances are unfeasible. The average computing time over the complete set of benchmark instances rises up to 1874 s. According to Corollary 1 when MOD0 is feasible, the optimal value of MOD1–MOD4 will be zero since there is an assignment of parking places that satisfies the time window requests of all the carriers. Conversely, if MOD0 is not feasible the optimal values to MOD1–MOD4 will be strictly positive. In this respect, the results in Table 11 confirm that, computationally, the outcome MOD0 gives valuable information not only with respect to MOD0 itself but also with respect to MOD1–MOD4. Indeed, all the 28 instances that are feasible for MOD0 were optimally solved by MOD1–MOD3, and 23 such instances were also optimally solved by MOD4. Furthermore, no model MODj, $j=1,\ldots,4$ was able to optimally solve any of the unfeasible instances within the maximum CPU time: neither the ones detected by CPLEX nor the ones detected by the unfeasibility check. On the other hand, by solving MOD1–MOD4, the feasibility/unfeasibility with respect to MOD0 was disclosed for only one instance with status *TL* (instance 47). This instance was optimally solved by all MOD1–MOD4 and is feasible with respect to MOD0 as its optimal value with respect to MOD1–MOD4 is zero.

MOD1, MOD2 and MOD4 always produced some solution, even if its optimality was not proven. In contrast, for seven instances MOD3 consumed the allowed computing time without finding a feasible solution. Recall that MOD3 can be unfeasible because of the maximum displacement constraint (21). Four of the instances without a solution for MOD3 (1, 5, 6, 44) are known to be unfeasible for MOD0, whereas for instances 18, 24 and 39 its condition with respect to MOD0 is unknown.

The average CPU times required by MOD1–MOD4 are considerably larger than those of MOD0, even if we restrict to the benchmark instances that terminate with a certificate of optimality. It seems however that the min–max objective of MOD2 is somehow less demanding than the sum-type objectives of MOD1, MOD3 and MOD4.

The effectiveness of MOD1–MOD4 is summarized in Table 2, where benchmark instances have been partitioned relative to their

status with respect to MOD0 in *Feasible*, *Unfeasible*, detected either by CPLEX or by the sufficient condition of Proposition 2, and *Time Limit*, when the time limit reached without knowing whether or not the instance is feasible. For each tested model we further partition each of the above groups of instances according to the possible outcomes of the tested model: optimum found (*OPT*), time limit reached with a feasible solution (*TL/S*), and time limit reached without a feasible solution (*TL/NS*). The entries in the table give the number of instances in each class and the average CPU times over the set of instances in the class. No clear conclusion can be drawn from Table 2 about the effectiveness of the models in terms of their capability for finding good quality solutions. For instances in *Feasible*, MOD4 performs slightly worse than the other models, since in 5 instances non-optimal solutions were found. But as already mentioned, in terms of their capability of proving feasibility, the other models work exactly as MOD0. For *Unfeasible* and *Time Limit* instances, the performance of MOD1, MOD2 and MOD4 is the same, and a little better than that of MOD3.

For the reasons explained before and justified by the Example A.1 of the Appendix, the experiments described above were run without taking into account the SECs (9). Still, we made some tests to confirm their potential usefulness empirically. First, we checked that, indeed, all 60 benchmark instances satisfied the condition that ensures that the LP relaxation of MOD0 will be feasible, even with the addition of the SECs (For a detailed statement of the condition see expression (39) in the Appendix). Note that, in turn, this means that the LP values of MOD1–MOD4 are all zero. Then, we implemented a callback with a separation procedure for the SECs and run a second set of experiments. The rationale for these additional tests was that the addition of the SECs could reinforce the LP relaxation of some nodes of the enumeration tree, even if they had no effect on the root node. In our second set of experiments, constraints (9) were separated at all the nodes of the enumeration tree with depth up to 15. This strategy was hopeless: while we appreciated no difference in the number of instances whose unfeasibility was proven, the number of instances that terminated with a certificate of optimality decreased notably for all five models MOD0–MOD4. Again, in all cases when optimality could not be proven the lower bound at termination was zero.

4.3. Cross-evaluation of solutions

In the rest of this section we further analyze models MOD1–MOD4. In particular, we evaluate the goodness of the solutions produced by each of the models relative to the other models. We restrict our analysis to the 36 instances, which were not solved to optimality by all models, and allow for a comparison among MOD1–MOD4. First we introduce some additional notation that we will use to present our results. For $j=1,\ldots4$, let z^j denote the objective function of model MODj, so for any feasible solution δ, $z^j(\delta)$ is the objective function value of solution δ for model MODj. Let also δ_{jk} be the best solution obtained with model MODj for instance k, when it found some feasible solution, with value $z^j(\delta_{jk})$. As we will see, for a given instance k, not always the best-known solution for MODj coincides with δ_{jk}, since in some cases it was obtained with a different model. Hence, we will also use the notation v_{jk} to denote the best-known value for MODj and instance k. That is

$$v_{jk} = \min_{i=1,\ldots,4}\{z^j(\delta_{ik})\}.$$

Furthermore for each model MODj and instance k we compute the percent deviations of the objective values of the best solutions produced by all four models, $z^j(\delta_{ik})$, $i=1,\ldots4$, with respect to the best-known values v_{jk}, which are denoted by Δ^j_{ik}. These deviations are computed as

$$\Delta^j_{ik} = 100\frac{z^j(\delta_{ik}) - v_{jk}}{(1+v_{jk})},$$

where a "1" has been added in the denominator to prevent dividing by zero.

Table 12 in the Appendix gives for each model MODj and instance k, its best-known value, v_{jk}, together with Δ^j_{ik}, $i=1,\ldots,4$. When MODi produced no solution for a given instance k within the allowed CPU time, the entries Δ^j_{ik} are empty for $j=1,\ldots,4$. Solutions from MOD1, MOD2, and MOD4 are always feasible and the solutions produced by one of them are also feasible for the other models. On the contrary, MOD3 can be unfeasible for some instances or, despite being feasible, the solutions produced by MOD1, MOD2, and MOD4 may not be feasible for it. When the solution produced by MODi for instance k was not feasible for MOD3, the entry Δ^3_{ik} is -1. An empty entry under the column v_{3k} indicates that no feasible solution is known for the corresponding instance for MOD3. The last two rows of Table 12 give average percentage gaps Δ^j_{ik}, over the set of all the instances k for which MODi obtained a feasible solution for MODj, and the total number of best-known solutions for MODj produced by MODi.

The information of Table 12 is summarized in Table 3. The entry in row MODi and column MODj gives the number of tested instances for which the solution produced MODi gives the best-known value for MODj, i.e. $v_{jk}=z^j(\delta_{ik})$. While for MOD1, MOD2, and MOD4 feasible solutions are known for all 36 tested instances, for MOD3 feasible solutions are known for only 28 instances. This is indicated in column under $|S^i|$. Row labeled *Total Best* gives the sum of the above rows. For each of the models, this is the overall number of best-known solutions obtained for this model, with any of the models MODi, $i=1,\ldots,4$. Indeed this value can be bigger than the number of instances tested for this model, $|S^j|$, when more than one model produced a best-known solution for some instance k. In the case of MOD3 this may also happen because some other model produced a feasible solution when MOD3 was not. This happened in particular with instance 1 for which MOD3 terminated with *TL/NS* but the solution produced by MOD2, δ_{21}, is feasible to MOD3. Column *ALL* gives the total number of best-known solutions produced by each of the models.

From the results presented in Tables 3 and 12, it is clear that each model usually produces the higher number of best-known solutions for it. MOD1, MOD2, MOD3 give best-known solutions for the rest of the models, while MOD4 finds very few best-known solutions for any of the other models. MOD3 performs in general better than any other model, giving even nearly as many number of best-known solutions for MOD1 as MOD1 itself, and competitive solutions for the rest of the models as indicated by the percent deviation gaps. This reinforces the interest of MOD3, which, on the other hand, can be seen as a good compromise from the modeling point of view, as it guarantees a maximum deviation from the preferred time window for all requests. Since the application framework that motivates the study of the PAP aims at producing

Table 3
Cross comparing models.

	$\lvert S^i\rvert$	MOD1	MOD2	MOD3	MOD4	ALL
MOD1	36	21	7	11	8	47
MOD2	36	5	29	6	5	45
MOD3	28	22	10	24	7	63
MOD4	36	0	0	0	28	28
Total Best		48	46	41	48	
No Solution		0	0	7	0	

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

"fair" solutions, the good performance of MOD3 with respect to the other models increases its relevance.

4.4. Sensitivity analysis of maximum displacement

MOD1 and MOD3 only differ in Constraints (21), which limit the maximum non-accomplishment per request in MOD3. In all the experiments described above the value of this parameter was set to $d=60$ min. In this section, we present a sensitivity analysis on the value of the maximum displacement per request. To see the effect of Constraints (21), MOD3 has been solved again with varying values of the maximum displacement parameter $d \in \{40, 50, 70, 80, \infty\}$. We use MOD3($d$) to refer to MOD3 with a parameter value d. Note that MOD3(∞)=MOD1. A summary of the obtained results is presented in Table 4. Rows correspond to varying values of d. In each case, we restrict to the subset of instances from the original set of 60 benchmarks for which MOD3 (d) terminated with a feasible solution, which is denoted by F^d. The number of such instances is given in the column under $|F^d|$. The next two columns indicate the number of instances for which the optimality of the obtained solution was proven (*#OPT*) and was not proven (*#TL/S*) within the maximum time limit. The last column under *quality index* (*qi*) has been computed as follows: for each instance $k \in F^d$ and parameter value d, we denote by v_k^d its objective function value for MOD3(d). Then, for each instance k, we compute $\overline{v}_k = \max_d\{v_k^d : k \in F^d\}$, where this maximum is computed over the set of models for which k has a feasible solution. That is, $\overline{v}_k$ is the worse outcome we have obtained with instance k over all models MOD3(d) for which $k \in F^d$. Usually, $\overline{v}_k = v_k^\infty$. Now, for each parameter value d and each instance $k \in F^d$ we compute the ratio $v_k^d/\overline{v}_k$, which is a measure of the quality of the solution obtained with MOD3(d) for instance k relative to the worse outcome of MOD3 for instance k for varying values of d. In other words, for a given instance k, $100(1-(v_k^d/\overline{v}_k))$ is the percentage improvement of model MOD3(d) with respect to its worse possible outcome for the varying values of d. Finally, for each parameter value d, we compute the index

$$qi(d) = 1 - \frac{1}{|F^d|}\sum_{k \in F^d}\frac{v_k^d}{\overline{v}_k} \tag{37}$$

which is an average measure of the quality of the solutions of model MOD3(d) when parameter d is used. These values are given in the last column of Table 4.

The average results presented in Table 4 show that, in general, models perform as expected. When the maximum deviation parameter increases the MOD3 finds more feasible solutions. However, the quality of the solutions obtained when the parameter is bigger deteriorates. For instance, MOD3(40) finds 44 solutions that, on average, are 75% better than the worse solution found. In contrast, MOD3(80) finds 56 solutions, which are only 59.4% better than the worse solution found. Finally, MOD1 provides always a solution but it is only 53.5% better than the worse solution found.

5. Heuristics

As we have seen CPLEX can be quite time consuming on some of the proposed formulations. Thus heuristic methods can be of interest, as potentially they could produce good solutions in smaller computing times. Next we present a simple heuristic to obtain feasible solutions to the PAP and compare its results with those produced by MOD0–MOD4 . The heuristic consists of a greedy constructive phase, followed by a simple local search.

For the constructive phase, first requests are ordered by non-decreasing values of their earliest start time, a_i. Ties are broken by non-decreasing values of their latest start time, b_i. If ties remain, they are broken by non-decreasing values of the requests durations s_i. Possible remaining ties are broken arbitrarily. Hence we assume that

$$\begin{cases} \text{(i)} & a_i \le a_{i+1}, \quad i \in Q; \\ \text{(ii)} & b_i \le b_{i+1} \quad \text{if } a_i = a_{i+1}; \\ \text{(iii)} & s_i \le s_{i+1} \quad \text{if } a_i = a_{i+1} \text{ and } b_i = b_{i+1}. \end{cases} \tag{38}$$

In the constructive phase parking places are considered in turn. When a parking place is selected, unassigned requests are explored by increasing order of their indices, and assigned to the current place provided that they preserve the feasibility of the current assignment to the parking place. Requests with time window conflict with the current assignment remain unassigned and will be considered for assignment to subsequent parking places.

When all parking places have been considered the constructive phase enters a final step. Now unassigned requests are considered in turn and assigned to some parking place outside their preferred time windows. Each such request is scheduled either at the very beginning or the very end of some parking place schedule, depending on the alternative which incurs the smallest earliness or tardiness. For a given request, the parking place is selected so as to minimize the resulting associated penalty. In this final step unassigned requests are considered by increasing values or their time duration, s_i. At the end of the algorithm all requests are assigned to some parking place. However, the assignment need not be feasible, as some requests may be scheduled outside their time windows. A pseudocode of the algorithm is presented in Algorithm Constructive. $A(p)$ contains the indices of all the requests assigned to parking place p. $Init_p$ and End_p, respectively, denote the starting times of the first and last requests assigned to parking place p.

In the local search we try to interchange the schedule of two requests. Again the index order of the requests is the one indicated by (38). We consider two different pairs of requests $i,j \in Q$, $i \neq j$: (i) i and j are adjacent in Q independent from the scheduled parking place p, (ii) i and j are not adjacent but in Q there is at most one request between i and j, i.e. $|i-j| \le 2$, again independent from the scheduled parking place. All possible interchanges are considered until no more improvements are possible.

Detailed results are given in Table 13 of the appendix, where for a better comparison some columns have been copied from Table 11. These are (i) the first five columns, which summarize the characteristics of the instances; (ii) Column *Type* which gives the status of MOD0 at termination of CPLEX; and, (iii) the five

Table 4
Comparing MOD1 and MOD3.

	d	$\lvert F^d\rvert$	#OPT	# TL/S	Quality index ($qi(d)$)
MOD3	40	44	30	14	0.750
	50	50	30	20	0.666
	60	52	29	23	0.631
	70	55	29	26	0.601
	80	56	29	27	0.594
	∞	60	29	31	0.535

Table 5
Summary of heuristic results.

	MOD0 Feas	MOD1 Val	MOD2 Val	MOD3		MOD4 Val
				Feas	Val	
CPLEX-b	20	48	46	24	44	29
Same	8	8	8	28	8	12
Heur-b	0	4	6	0	0	19
%*Gpap*		4,706.99	1,545.86		6,177.04	74.99

columns labeled *SOL*, which give the value of the best solution produced by each model. Columns labeled *C-HEUR* give the objective values obtained with the constructive phase of the heuristic. Empty entries in column *C-HEUR* of MOD0 or MOD3 indicate that the heuristic was not able to find a feasible solution for the corresponding instance. Columns labeled *I-HEUR* give the values obtained with the local search, when it improved the solution of the constructive phase. Empty entries corresponding to instances when the constructive heuristic found a feasible solution indicate that the local search was not able to improve the solution of the heuristic phase.

A summary of the results is given in Table 5. Entries in rows CPLEX-*b*, *Same* and *Heur-b* of Table 5 give, for each model, the number of instances when CPLEX outperformed the heuristic, both methods produced the same output, and the heuristic outperformed CPLEX, respectively. For MOD0 and MOD3 the performance was measured in terms of the number of feasible solutions found (columns under *Feas*). The entries in row CPLEX-*b* of these two columns indicate the number of instances for which only CPLEX produced a feasible solution, whereas the entries in row *Same* indicate the number of instances for which both CPLEX and the heuristic found a feasible solution. For MOD1–MOD4 the performance was measured in terms of the objective function values (columns under *Val*). Note that MOD3 has been compared for both measures. The last row, %*Gpap*, gives the average percentage deviation gaps of the solutions obtained with the heuristic from the solutions obtained with CPLEX. For each model and each instance, the gaps have been computed as $100(z_{heur}-z_{Mi})/z_{Mi}$, where z_{Mi} and z_{heur}, respectively, denote the objective function value of the solutions produced, for the corresponding instance, by CPLEX with MOD-*i* and by the heuristic. For each model, the average has been computed over the set of instances for which both CPLEX and the heuristic produced a feasible solution.

As could be expected, in general, the quality of the solutions produced by CPLEX is notably better than that of the heuristic solutions for each of the considered models. This can be appreciated both by the number of instances where CPLEX outperforms the heuristic and by the average percentage deviation gaps in row %*Gpap*.

While CPLEX found a feasible solution for MOD0 for 28 instances, the heuristic solution was feasible for MOD0 for only eight such instances. This is an indicator of the difficulty for finding feasible solutions with no deviation with respect to the requested time windows even if they exist. A similar behavior can be observed for MOD3, where the heuristic produced a solution within the maximum allowed in MOD3 for only 28 instances,

Algorithm 1. Constructive phase.

```
Constructive Phase
  Input: c, Q, a, b, s,
Initialize list UNASSIGNED with elements i ∈ Q in non-decreasing order of a_i with
  ties broken as explained in (38);
Initialize list of requests assigned to each parking place: A(p) ← ∅,  p = 1 ... c
for (p ← 1 ... c) do
    i ← FIRST(UNASSIGNED);
    Init_p ← a_i
    t ← Init_p
    while (i ≠ nil) do
        A(p) ← A(p) ∪ {i}        (Assign request i to parking place p)
        t ← max(t + s_i, a_i)
        UNASSIGNED ← UNASSIGNED \ {i}
        i ← NEXT(UNASSIGNED)
        while (b_i < t and i ≠ nil) do
            i ← NEXT(UNASSIGNED)
        end
    end
    End_p ← t
end
Sort elements of UNASSIGNED by non-decreasing values of s_i
while ( UNASSIGNED ≠ ∅) do
    i ← FIRST(UNASSIGNED);
    for (p ← 1 ... c) do
        δ_p ← Min{|a_i − Init_p|, |b_i − End_p|}
    end
    p̄ ← argmin_{p=1...c}{δ_p}
    A(p̄) ← A(p̄) ∪ {i}
    Update Init_p̄, or End_p̄ as appropriate
end
```

whereas this number rises up to 52 for CPLEX. Furthermore in all the instances where the heuristic solution was feasible for MOD3, the solution of CPLEX outperformed it.

Nevertheless, for some instances the heuristic solution was better than that produced by CPLEX for some model. This happens particularly with MOD4, where the heuristic solution was better than the solution obtained with CPLEX for 19 instances. We attribute the poor performance of CPLEX with MOD4 to the symmetry of its objective function. Note that in MOD4 the objective just counts the number of requests scheduled outside their preferred time window. This means that many alternative optimal solutions may exist and thus, pruning nodes in the enumeration tree may become an extremely arduous task. On the other hand, as can be seen in Table 13 and as indicated by the large average percentage gaps in Table 5, in most cases the heuristic solutions that outperformed those of CPLEX for MOD4 were not good for the other models. This is consistent with the results of Tables 3 and 12 which, as we have already noted, indicate that good solutions for MOD4 tend to be of low quality for the other models. This observation is now particularly true in what refers to MOD3, for which none of the heuristic solutions which were good for MOD4 were even feasible. The computing times required by the heuristic are, in general, small. This is not surprising giving its simplicity. They are on average 0.003 s for the first phase and 0.03 s for the improvement phase.

6. Conclusions

In this work we have presented the Parking Slot Assignment Problem (PAP), a novel problem motivated by the need of providing parking space to carriers for their loading/unloading operations. The PAP is to find assignments of carriers to parking places that satisfy their time window requests. We have studied modeling alternatives for the PAP, including a feasibility formulation, which looks for an assignment satisfying all the carriers' time window requests, and several other models, which allow deviations from the requested time windows, that evaluate the degree of non-accomplishment with different criteria.

The domains of the proposed formulations have been compared and an unfeasibility sufficient condition for the feasibility mode has been given, which has proven to be very effective in practice.

We have evaluated and compared experimentally the proposed models by solving a set of test instances using CPLEX. In our experiments the computing time has been limited to one hour, to avoid very long runs, but also to adapt to the nature of the problem which demands a solution daily. The obtained results have been presented together with a detailed analysis and comparison of various indicators, including a sensitivity analysis on the maximum allowed displacement from the requested time window for one of the models. Broadly speaking these results indicate that the earliness/tardiness criterion tends to produce solutions which are also good for the rest of the models.

Since CPLEX can be quite time consuming on some of the proposed formulations, we have proposed a simple heuristic and compared its solutions with those obtained with CPLEX. As could be expected, in general, the quality of the solutions produced by CPLEX is considerably better than that of the heuristic solutions. Nevertheless, for MOD4 the heuristic solution outperformed CPLEX for about one-third of the instances. We attribute this to the objective function which may produce very many alternative optima.

A promising avenue for research is to further explore some of the more general models that we have also proposed, which include weights in the terms of the objective function. Another avenue of research is developing more sophisticated heuristics that may produce good quality solutions in small computing times. On the other hand, incorporating uncertainty to our data will certainly lead to more realistic models. Finally, the proposed formulations can be exploited for other similar transportation problems of resource allocation with time windows, like the administration of public rechargeable points for electrical vehicles or the use of docks in a freight terminal.

Acknowledgments

This research has been partially supported by the Spanish Ministry of Economy and Competitiveness through projects TRA2009-14759-C02-01 and MTM2012-36163-C06-05, and ERDF funds. This support is gratefully acknowledged.

Appendix A

A.1. Example

A feasible solution to the LP relaxation of the domain defined by constraints (2)–(8) plus the SEC constraints (9) can be obtained in the following ways (see Fig. 3):

- The flow through arcs connecting the depot with each other node (dashed arcs in Fig. 3) is c/q in both directions, i.e. $x_{v_d j} = x_{i v_d} = c/q,\ \forall i,j \in Q$.
- The flow through any other arc connecting two nodes $i,j \in Q$ (solid lines in Fig. 3) is $\frac{q-c}{q(q-1)}$ in both directions, i.e. $x_{ij} = x_{ji} = \frac{q-c}{q(q-1)},\ \forall i,j \in Q$.
- All time variables are set to the upper end of their time window, i.e. $t_i = b_i,\ \forall i \in Q$.

By construction it is clear that flow constraints (3) are satisfied at the depot and at the rest of the nodes. To see that constraints (9) are always satisfied, let S be a subset of $Q/\{v_d\}$ with $|S| = r$. Since each of the arcs with both endnodes in S has value $\frac{q-c}{q(q-1)}$, the left-hand side of the associated constraint (9) is $r(r-1)\frac{q-c}{q(q-1)}$. Thus the constraint is satisfied if and only if $r(r-1)\frac{q-c}{q(q-1)} \le r-1$, which always holds since $r/q \le 1$ and $\frac{q-c}{q-1} \le 1$.

Finally, Constraints (5) are satisfied if

$$b_j \ge b_i + s_i - M\left(1 - \frac{q-c}{q(q-1)}\right). \tag{39}$$

We have checked that with the data used in the computational section $1-(q-c)/[q(q-1)] \ge 0.95$. Thus, if we use the tightest value for M, which is $M_{ij} = b_i + s_i - a_j$, the above condition is satisfied when $b_j \ge 0.95a_j + 0.05(b_i + s_i)$, which is intuitively true in most cases. In particular, this condition is always fulfilled by all the realistic data we have used in the computational section.

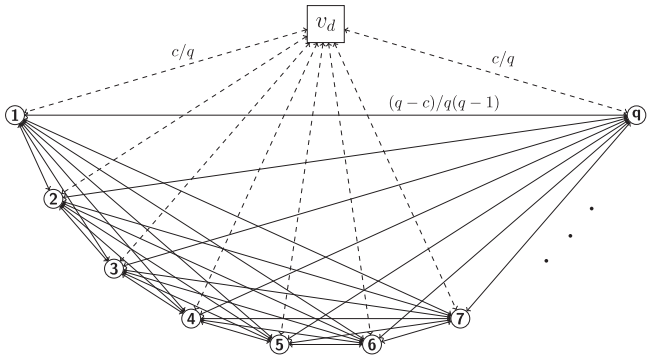

Fig. 3. Feasible solution for LP relaxation of Constraints (2)–(8).

Table 6
Parameters and sets of the models.

Parameters and sets	Description
$[0, T]$	Time interval for the assignment of parking slots
c	Number of parking places for vehicles to load/unload goods
Q	Index set of requests
q	Number of requests ($q = \|Q\|$)
a_i	Earliest starting time for request $i \in Q$
b_i	Latest starting time for request $i \in Q$
s_i	Duration of operation $i \in Q$
v_d	Fictitious depot
V	Set of vertices for the fictitious routes: $V = Q \cup \{v_d\}$
A	Set of arcs for the fictitious routes
d	Maximum allowed displacement from requested time window in MOD4
M	Big-M used in Constraints (5) relating variables
K	Big-M used in Constraints (24) and (25) of MOD3
w_i	Weight of request $i \in Q$ in the objective function of MOD1 and MOD3
μ_t	Unit cost for outsourced parking place at time slot $t \in \{0, \ldots, T-1\}$
κ	Constant value of objective function in MOD0

Table 7
Objective functions.

Model		Objective function	
0	Feasibility	$z^0(x,t) = \kappa$	
1	Min ET	$z^1(x,t) = \sum_{i \in Q} w_i e_i$	(13)
2	Min–Max ET (a&b)	$z^2(x,t) = m$	(15)
3	Min ET st Max	$z^3(x,t) = \sum_{i \in Q} w_i e_i$	(20)
4	Min Num (a&b)	$z^4(x,t) = \sum_{i \in Q} \beta_i$	(22)
4′	Min Num weight(a&b)	$z^{4'}(x,t) = \sum_{i \in Q} s_i \beta_i$	(26)
5	Cost function	$z^5(x,y,t) = \sum_{t=0}^{T} n_t$	(33)
5′	Cost function time dependent	$z^{5'}(x,y,t) = \sum_{t=0}^{T} w_t \sum_{i \in Q} O_{it}$	(36)

Table 8
Decision variables.

MOD	Common variables	Description
All	x_{ij}	If request j is performed exactly after request i (binary)
All	t_i	The time when request i begins being served (continuous)
MOD	**Specific variables**	**Description**
1/2/3/4	e_i	Earliness/tardiness associated with request i (continuous)
2/2b	m	Maximum earliness/tardiness (continuous)
4/4b	β_i	Indicator for request i if performed on time (binary)
5/5b	n_t	Number of outsourced requests scheduled at time slot t (integer)
5/5b	y_{ij}	If request j is performed exactly after request i in outsourced parking space (binary)
5b	O_{it}	Indicator for request i occupying an outsourced space at time slot t (binary)

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

Table 9
Constraints of the models.

MOD		**Common constraints**			
All		Parking places	(2)	$\sum_{j \in Q} x_{v_d j} \le c$	
All		Connectivity	(3)	$\sum_{(i,j) \in A} x_{ij} - \sum_{(j,i) \in A} x_{ji} = 0$	$i \in Q$
All		Request completion	(4)	$\sum_{(i,j) \in A} x_{ij} = 1$	$i \in Q$
All		Time	(5)	$t_i + s_i - t_j \le (1 - x_{ij})M$	$i,j \in Q,\ (i,j) \in A$
All		Domain t	(7)	$0 \le t_i \le T$	$i \in Q$
All		Domain x	(8)	$x_{ij} \in \{0,1\}$	$(i,j) \in A$
None		Subtour elimination	(9)	$\sum_{\substack{(i,j) \in A \\ i,j \in W}} x_{ij} \le \lvert W \rvert - 1$	$W \subset V$
	MOD	**Particular constraints**			
0	Feasibility	Time window	(6)	$a_i \le t_i \le b_i$	$i \in Q$
1	Min ET	ET	(10)	$e_i \ge a_i - t_i$	$i \in Q$
		”	(11)	$e_i \ge t_i - b_i$	$i \in Q$
		”	(12)	$e_i \ge 0$	$i \in Q$
2	Min-Max ET (a)	ET +			
		MET	(16)	$m \ge e_i$	$i \in Q$
2	Min-Max ET (b)	MaxET	(17)	$m \ge a_i - t_i$	$i \in Q$
		”	(18)	$m \ge t_i - b_i$	$i \in Q$
		”	(19)	$m \ge 0$	
3	Min ET st Max	ET +			
		Maximum displacement	(21)	$e_i \le d$	$i \in Q$
4	Min Num (a)	ET +			
		Num	(23)	$\beta_i K \ge e_i$	$i \in Q$
4	Min Num (b)	NumET	(24)	$\beta_i K \ge a_i - t_i$	$i \in Q$
		”	(25)	$\beta_i K \ge t_i - b_i$	$i \in Q$
5	Cost (a)	Connectivity y	(29)	$\sum_{(i,j) \in A} y_{ij} - \sum_{(j,i) \in A} y_{ji} = 0$	$i \in Q$
		Request assignment y	(30)	$\sum_{(i,j) \in A} (x_{ij} + y_{ij}) = 1$	
		Time y	(31)	$t_i + s_i - t_j \le (1 - y_{ij})M$	$i,j \in Q,\ (i,j) \in A$
		Domain y	(32)	$y_{ij} \in \{0,1\}$	$(i,j) \in A$
5	Cost (b)	Cost (a) +			
		Outsourcing	(34)	$O_{it'} \ge O_{it} - (1 - y_{ij})$	$i,j \in Q, t, t' \in [0,T]$ with $t' = t + s_i$
		Outsourcing (b)	(35)	$O_{it} \le O_{it'}$	$t \in [0,T], t+1 \le t' \le t + s_i$

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

Table 10
Variable and constraint count.

		MOD0	MOD1	MOD2		MOD3	MOD4		MOD5		
		Feasibility	Min ET	Min-Max ET		Min ET st Max	Min Num		Outsource		
				a	b		a	b	a	b	
Variables		q^2+2q	q^2+3q	q^2+3q+1	q^2+2q+1	q^2+3q	q^2+4q	q^2+3q	q^2+3q	$q^2+3q+qT$	
Common constraints											Num Const
Parking places	(2)	X	X	X	X	X	X	X	X	X	1
Connectivity	(3)	X	X	X	X	X	X	X	X	X	q
Request completion	(4)	X	X	X	X	X	X	X			q
Time	(5)	X	X	X	X	X	X	X	X	X	$q(q-1)$
Domain t	(7)	X	X	X	X	X	X	X	X	X	$[2q]$
Domain x	(8)	X	X	X	X	X	X	X	X	X	$[2q(q-1)]$
Particular Constraints											
Time window	(6)	X							X	X	$2q$
ET	(10)–(12)		X	X		X	X				$2q$
MET	(16)			X							q
MaxET	(17) and (18)				X						$2q$
Maximum displacement	(21)					X					q
Num	(23)						X				q
NumET	(24) and (25)							X			$2q$
Connectivity y	(29)								X	X	q
Request y	(30)								X	X	q
Time y	(31)								X	X	q
Domain y	(32)								X	X	$[2q]$
Outsource	(34)									X	qT
Outsourceb	(35)									X	$qT\sum_i s_i$
Constraints		q^2+3q+1	q^2+4q+1	q^2+5q+1	q^2+4q+1	q^2+5q+1	q^2+5q+1	q^2+3q+1	$2q^2+3q+1$	$q^2(2+T)+1+q(3-T+T\sum_i s_i)$	

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

Table 11
Results of MOD0–MOD4 in the set of instances.

Id	Rq	c	Hop	γ	MOD0			MOD1		MOD2		MOD3		MOD4	
					Type	SOL	TIME	SOL	TIME	SOL	TIME	SOL	TIME	SOL	TIME
1	111	7	6	2.64	UC	−2	3,600	1,352	3,600	48	3,600		3,600	30	3,600
2	75	4	6	3.13	UC	−2	3,600	1,391	3,600	54	3,600	1,359	3,600	24	3,600
3	60	4	6	2.50	TL		3,600	200	3,600	12	3,600	142	3,600	9	3,600
4	58	6	6	1.61	F	0	1	0	61	0	29	0	32	0	32
5	119	6	6	3.31	UC	−2	3,600	2,602	3,600	73	3,600		3,600	55	3,600
6	82	4	6	3.42	UC	−2	3,600	1951	3,600	71	3,600		3,600	26	3,600
7	92	5	6	3.07	TL		3,600	629	3,600	28	3,600	675	3,600	16	3,600
8	42	3	6	2.33	F	0	466	0	3,248	0	1,682	0	2,017	2	3,600
9	48	5	6	1.60	F	0	5	0	510	0	49	0	46	0	122
10	73	6	6	2.03	F	0	9	0	933	0	187	0	555	0	420
11	66	6	6	1.83	F	0	1	0	97	0	146	0	392	0	154
12	74	4	6	3.08	UC	−2	3,600	1,169	3,600	44	3,600	1,111	3,600	19	3,600
13	64	5	6	2.13	F	0	45	0	3,573	0	489	0	3,373	2	3,600
14	36	3	6	2.00	F	0	2	0	41	0	9	0	8	0	18
15	24	2	6	2.00	U	−1	394	13	3,600	3	3,600	13	3,600	1	3,600
16	87	6	6	2.42	TL		3,600	428	3,600	19	3,600	409	3,600	13	3,600
17	104	8	6	2.17	TL		3,600	253	3,600	15	3,600	245	3,600	14	3,600
18	146	8	6	3.04	TL		3,600	1,885	3,600	76	3,600		3,600	49	3,600
19	76	7	6	1.81	F	0	156	0	3,261	0	720	0	422	0	1,703
20	30	3	6	1.67	F	0	0	0	6	0	9	0	2	0	3
21	69	7	6	1.64	F	0	7	0	65	0	124	0	94	0	62
22	76	6	6	2.11	F	0	26	0	3,304	0	543	0	3,258	0	1,437
23	136	8	6	2.83	TL		3,600	1,446	3,600	71	3,600		3,600	34	3,600
24	54	3	6	3.00	UC	−2	3,600	495	3,600	35	3,600	531	3,600	11	3,600
25	57	4	6	2.38	F	0	11	0	3,369	0	2,651	0	3,291	1	3,600
26	82	5	6	2.73	TL		3,600	995	3,600	44	3,600	885	3,600	19	3,600
27	75	8	6	1.56	F	0	2	0	367	0	73	0	222	0	482
28	45	3	6	2.50	TL		3,600	9	3,600	3	3,600	13	3,600	2	3,600
29	98	7	6	2.33	TL		3,600	12	3,600	14	3,600	90	3,600	7	3,600
30	21	2	6	1.75	U	−1	1,022	7	3,600	2	3,600	7	3,600	1	3,600
31	72	4	10	1.80	F	0	2,074	0	3,313	0	190	0	1,402	2	3,600
32	109	4	10	2.73	TL		3,600	1166	3,600	44	3,600	829	3,600	36	3,600
33	50	2	10	2.50	TL		3,600	307	3,600	33	3,600	306	3,600	10	3,600
34	106	7	10	1.51	F	0	3	0	971	0	714	0	223	0	805
35	129	5	10	2.58	TL		3,600	1,431	3,600	221	3,600	1,145	3,600	40	3,600
36	62	2	10	3.10	UC	−2	3,600	730	3,600	42	3,600	652	3,600	18	3,600
37	51	3	10	1.70	F	0	2	0	71	0	22	0	7	0	132
38	137	6	10	2.28	TL		3,600	529	3,600	205	3,600	352	3,600	20	3,600
39	97	3	10	3.23	UC	−2	3,600	1,764	3,600	69	3,600		3,600	38	3,600
40	135	5	10	2.70	TL		3,600	1,989	3,600	121	3,600	1,531	3,600	60	3,600
41	55	2	10	2.75	TL		3,600	361	3,600	41	3,600	357	3,600	11	3,600
42	170	7	10	2.43	TL		3,600	1573	3,600	317	3,600	1217	3,600	65	3,600
43	100	4	10	2.50	TL		3,600	274	3,600	22	3,600	279	3,600	18	3,600
44	220	7	10	3.14	UC	−2	3,600	21,537	3,600	438	3,600		3,600	168	3,600
45	237	8	10	2.96	TL		3,600	27,701	3,600	419	3,600		3,600	186	3,600
46	77	7	6	1.83	F	0	23	0	333	0	258	0	323	0	618
47	46	4	6	1.92	TL		3,600	0	839	0	98	0	459	0	2,010
48	22	2	6	1.83	F	0	0	0	2	0	2	0	1	0	2
49	23	2	6	1.92	F	0	1	0	21	0	3	0	3	0	5
50	32	3	6	1.78	F	0	2	0	86	0	6	0	8	0	68
51	31	3	6	1.72	F	0	0	0	16	0	3	0	4	0	4
52	29	3	6	1.61	F	0	1	0	16	0	3	0	4	0	4
53	23	2	6	1.92	F	0	1	0	34	0	4	0	11	0	45
54	60	5	6	2.00	F	0	7	0	142	0	68	0	208	0	53
55	41	4	6	1.71	F	0	3	0	69	0	23	0	17	0	23
56	116	6	10	1.93	F	0	115	0	3,385	0	3,281	0	623	1	3,600
57	103	6	10	1.72	F	0	58	0	1,329	0	3,264	0	3,287	0	3,322
58	66	4	10	1.65	F	0	4	0	173	0	71	0	17	0	270
59	73	4	10	1.83	TL		3,600	2	3,600	2	3,600	3	3,600	1	3,600
60	151	8	10	1.89	TL		3,600	286	3,600	194	3,600	273	3,600	17	3,600

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

Table 12
Percent deviations from best-known solutions.

Id	Type	MOD1					MOD2					MOD3					MOD4				
		Best	Δ^1_{1k}	Δ^1_{2k}	Δ^1_{3k}	Δ^1_{4k}	Best	Δ^2_{1k}	Δ^2_{2k}	Δ^2_{3k}	Δ^2_{4k}	Best	Δ^3_{1k}	Δ^3_{2k}	Δ^3_{3k}	Δ^3_{4k}	best	Δ^4_{1k}	Δ^4_{2k}	Δ^4_{3k}	Δ^4_{4k}
1	UC	1,352	0.00	72.95		1,009.53	48.0	108.16	0.00		1,516.33	2,339	−1	0.00		−1	30	74.19	141.94		0.00
2	UC	1,359	2.35	52.13	0.00	541.69	54.5	166.67	0.00	9.91	1,307.21	1,359	−1	52.13	0.00	−1	24	88.00	144.00	116.00	0.00
3	TL	142	40.56	119.58	0.00	479.02	12.0	200.00	0.00	123.08	938.46	142	40.56	119.58	0.00	−1	9	110.00	250.00	110.00	0.00
5	UC	2,602	0.00	60.68		456.40	73.5	124.16	0.00		827.52	−1	−1	−1		−1	55	33.93	78.57		0.00
6	UC	1,951	0.00	57.79		264.60	71.0	108.33	0.00		970.83	−1	−1	−1		−1	26	70.37	159.26		0.00
7	TL	629	0.00	65.63	7.30	949.21	28.5	133.90	0.00	93.22	2042.37	675	−1	54.36	0.00	−1	16	105.88	252.94	147.06	0.00
8	F	0	0.00	0.00	0.00	78,400.00	0.0	0.00	0.00	0.00	66,000.00	0	0.00	0.00	0.00	−1	0	0.00	0.00	0.00	200.00
12	UC	1,111	5.22	39.12	0.00	680.67	44.0	128.89	0.00	35.56	1,480.00	1111	−1	39.12	0.00	−1	19	110.00	185.00	135.00	0.00
13	F	0	0.00	0.00	0.00	18,700.00	0.0	0.00	0.00	0.00	10,300.00	0	0.00	0.00	0.00	−1	0	0.00	0.00	0.00	200.00
15	U	13	0.00	28.57	0.00	3,907.14	3.5	100.00	0.00	166.67	12,366.67	13	0.00	28.57	0.00	−1	1	100.00	250.00	100.00	0.00
16	TL	409	4.63	78.54	0.00	1,479.02	19.5	153.66	0.00	168.29	4,100.00	409	4.63	78.54	0.00	−1	13	128.57	292.86	114.29	0.00
17	TL	245	3.25	138.01	0.00	1,062.60	15.5	160.61	0.00	118.18	4,875.76	245	3.25	138.01	0.00	−1	14	73.33	286.67	53.33	0.00
18	TL	1,885	0.00	200.48		570.20	75.5	20.26	0.00		999.35	−1	−1	−1		−1	49	48.00	142.00		0.00
23	TL	1,446	0.00	259.78		634.35	70.5	17.48	0.00		1,132.17	−1	−1	−1		−1	34	77.14	222.86		0.00
24	UC	495	0.00	70.77	7.26	641.33	35.0	144.44	0.00	66.67	1,538.89	531	−1	59.21	0.00	−1	11	125.00	258.33	116.67	0.00
25	F	0	0.00	0.00	0.00	62,000.00	0.0	0.00	0.00	0.00	62,000.00	0	0.00	0.00	0.00	−1	0	0.00	0.00	0.00	100.00
26	TL	885	12.42	91.99	0.00	381.15	44.0	55.56	0.00	22.22	762.22	885	−1	91.99	0.00	−1	19	115.00	205.00	100.00	0.00
28	TL	9	0.00	280.00	40.00	16,510.00	3.0	25.00	0.00	250.00	20,925.00	9	0.00	280.00	40.00	−1	1	100.00	700.00	0.00	50.00
29	TL	12	0.00	3,676.92	600.00	20,207.69	3.0	0.00	287.50	325.00	20,175.00	12	0.00	3,676.92	600.00	−1	7	0.00	512.50	87.50	0.00
30	U	7	0.00	100.00	0.00	962.50	2.0	66.67	0.00	133.33	2,733.33	7	0.00	100.00	0.00	−1	1	50.00	400.00	50.00	0.00
31	F	0	0.00	0.00	0.00	42,900.00	0.0	0.00	0.00	0.00	33,900.00	0	0.00	0.00	0.00	−1	0	0.00	0.00	0.00	200.00
32	TL	829	40.60	155.18	0.00	1,140.84	44.5	73.63	0.00	34.07	2,143.96	829	−1	155.18	0.00	−1	36	51.35	91.89	29.73	0.00
33	TL	306	0.33	162.05	0.00	307.17	32.5	120.90	0.00	82.09	562.69	306	−1	162.05	0.00	−1	10	0.00	245.45	27.27	0.00
35	TL	1,145	24.96	1,088.70	0.00	752.88	60.0	32.79	263.11	0.00	1524.59	1,145	−1	−1	0.00	−1	40	51.22	187.80	46.34	0.00
36	UC	652	11.94	101.84	0.00	828.64	41.5	76.47	0.00	29.41	2,325.88	652	−1	101.84	0.00	−1	18	78.95	168.42	78.95	0.00
38	TL	352	50.14	3,858.36	0.00	1159.77	46.0	21.28	337.23	0.00	1,146.81	352	50.14	−1	0.00	−1	20	61.90	457.14	38.10	0.00
39	UC	1,764	0.00	82.27		606.12	69.5	84.40	0.00		965.25	−1	−1	−1		−1	38	43.59	97.44		0.00
40	TL	1,531	29.90	427.42	0.00	1,371.21	60.0	44.26	100.00	0.00	1,595.08	1,531	−1	−1	0.00	−1	60	29.51	90.16	14.75	0.00
41	TL	357	1.12	172.63	0.00	863.69	41.0	71.43	0.00	45.24	1,733.33	357	−1	172.63	0.00	−1	11	41.67	241.67	16.67	0.00
42	TL	1,217	29.23	1,853.90	0.00	1,736.12	60.0	78.69	420.49	0.00	1,503.28	1,217	−1	−1	0.00	−1	64	10.77	143.08	0.00	1.54
43	TL	274	0.00	249.82	1.82	3,239.64	22.5	138.30	0.00	129.79	4,202.13	274	0.00	249.82	1.82	−1	18	10.53	210.53	21.05	0.00
44	UC	21,537	0.00	117.26		191.88	438.0	33.03	0.00		143.96	−1	−1	−1		−1	160	0.00	29.19		4.97
45	TL	27,701	0.00	54.54		210.83	418.5	23.96	0.00		152.92	−1	−1	−1		−1	186	6.42	16.04		0.00
56	F	0	0.00	0.00	0.00	4,900.00	0.0	0.00	0.00	0.00	4,900.00	0	0.00	0.00	0.00	4900.00	0	0.00	0.00	0.00	100.00
59	TL	2	0.00	2,033.33	33.33	13,266.67	1.0	0.00	50.00	50.00	19,950.00	2	0.00	2033.33	33.33	−1	0	50.00	1,650.00	50.00	0.00
60	TL	273	4.74	6,191.06	0.00	1,295.62	45.0	19.57	322.83	0.00	1,554.35	273	4.74	−1	0.00	−1	0	72.22	700.00	66.67	0.00
Avrg			7.26	609.48	24.63	7,905.78		70.35	49.48	67.24	8,202.65		6.46	316.39	24.11	4900.00		53.27	244.74	54.26	23.79
No. Best			21	5	22	0		7	29	10	0		11	6	24	0		8	5	7	28

Table 13
Solution obtained with proposed heuristic.

Id	Rq	c	Hop	γ	MOD0			MOD1			MOD2			MOD3			MOD4		
					Type	SOL	C_HEUR	SOL	C_HEUR	I_HEUR	SOL	C_HEUR	I_HEUR	SOL	C_HEUR	I_HEUR	SOL	C_HEUR	I_HEUR
1	111	7	6	2.64	UC	−2		1,352	1,360		48	80					30	24	
2	75	4	6	3.13	UC	−2		1,391	1,610		54	109		1,359			24	21	
3	60	4	6	2.50	TL			200	652		12	97		142			9	9	
4	58	6	6	1.61	F	0	0	0	0		0	0		0	0		0	0	
5	119	6	6	3.31	UC	−2		2,602	2,577		73	135					55	36	
6	82	4	6	3.42	UC	−2		1,951	2,402		71	132					26	26	
7	92	5	6	3.07	TL			629	1,435		28	115		675			16	17	
8	42	3	6	2.33	F	0		0	361		0	95		0			2	5	
9	48	5	6	1.60	F	0		0	87		0	28		0	87		0	5	
10	73	6	6	2.03	F	0		0	97		0	53	45	0	97		0	2	
11	66	6	6	1.83	F	0	0	0	0		0	0		0	0		0	0	
12	74	4	6	3.08	UC	−2		1,169	1,476		44	125		1,111			19	18	
13	64	5	6	2.13	F	0		0	366		0	57		0	366		2	9	
14	36	3	6	2.00	F	0		0	84		0	52		0	84		0	2	
15	24	2	6	2.00	U	−1		13	182		3	73	55	13			1	3	
16	87	6	6	2.42	TL			428	830		19	92		409			13	14	
17	104	8	6	2.17	TL			253	309		15	69		245			14	11	
18	146	8	6	3.04	TL			1,885	2,048		76	98					49	33	
19	76	7	6	1.81	F	0		0	52	44	0	29	28	0	52	44	0	2	
20	30	3	6	1.67	F	0	0	0	0		0	0		0	0		0	0	
21	69	7	6	1.64	F	0	0	0	0		0	0		0	0		0	0	
22	76	6	6	2.11	F	0		0	136	81	0	75	39	0		162	0	2	
23	136	8	6	2.83	TL			1,446	1,435		71	100					34	27	
24	54	3	6	3.00	UC	−2		495	997		35	115		531			11	12	
25	57	4	6	2.38	F	0		0	314		0	82		0			1	6	
26	82	5	6	2.73	TL			995	1,218		44	104		885			19	19	
27	75	8	6	1.56	F	0	0	0	0		0	0		0	0		0	0	
28	45	3	6	2.50	TL			9	238		3	75	53	13		392	2	5	
29	98	7	6	2.33	TL			12	358		14	68		90			7	8	
30	21	2	6	1.75	U	−1		7	99		2	51	49	7	99		1	2	
31	72	4	10	1.80	F	0		0	336		0	84	81	0			2	6	
32	109	4	10	2.73	TL			1,166	2,436		44	149		829			36	27	
33	50	2	10	2.50	TL			307	843		33	147		306			10	9	
34	106	7	10	1.51	F	0	0	0	0		0	0		0	0		0	0	
35	129	5	10	2.58	TL			1,431	2,493		221	125		1,145			40	30	
36	62	2	10	3.10	UC	−2		730	1,683		42	164		652			18	18	
37	51	3	10	1.70	F	0		0	32		0	32	18	0	32		0	1	
38	137	6	10	2.28	TL			529	1,110		205	107		352			20	17	
39	97	3	10	3.23	UC	−2		1,764	3,294		69	182					38	28	
40	135	5	10	2.70	TL			1,989	2,787		121	147		1,531			60	34	
41	55	2	10	2.75	TL			361	933		41	156		357			11	10	
42	170	7	10	2.43	TL			1,573	2,333		317	110		1,217			65	32	
43	100	4	10	2.50	TL			274	1,098		22	137		279			18	11	
44	220	7	10	3.14	UC	−2		21,537	5,313		438	164					168	56	
45	237	8	10	2.96	TL			27,701	5,171		419	154					186	59	
46	77	7	6	1.83	F	0		0	21	1	0	21	1	0	21	1	0	1	
47	46	4	6	1.92	TL			0	75		0	36	32	0	75		0	3	
48	22	2	6	1.83	F	0	0	0	0		0	0		0	0		0	0	
49	23	2	6	1.92	F	0		0	97	48	0	54	18	0	97	48	0	2	
50	32	3	6	1.78	F	0		0	57	35	0	31	23	0	57	35	0	2	
51	31	3	6	1.72	F	0		0	65	51	0	34	24	0	65	51	0	2	
52	29	3	6	1.61	F	0		0	102		0	57	45	0	102		0	2	
53	23	2	6	1.92	F	0		0	90		0	34		0	90		0	3	
54	60	5	6	2.00	F	0		0	45		0	45	25	0	45		0	1	
55	41	4	6	1.71	F	0		0	7		0	7		0	7		0	1	
56	116	6	10	1.93	F	0		0	217		0	62	42	0		245	1	5	
57	103	6	10	1.72	F	0		0	120		0	45		0	120		0	4	
58	66	4	10	1.65	F	0	0	0	0		0	0		0	0		0	0	
59	73	4	10	1.83	TL			2	422		2	100	70	3			1	8	
60	151	8	10	1.89	TL			286	496		194	69		273			17	10	

References

[1] Ma L. Analysis of unloading goods in urban streets and required data. In: City logistics II. Japan: Institute of Systems Science Research; 2001. p. 367–79.
[2] Rosenhead J. Operational research in urban planning. Omega 1981;9:345–64.
[3] Gerra M, Ross M. An interactive city planning model. Omega 1973;1:747–55.
[4] Odani M, Tsuji T. An experiment to demonstrate the effectiveness of on-street parking facilities for delivery vehicles. In: City logistics II. Japan: Institute of Systems Science Research; 2001. p. 351–65.
[5] Muñuzuri J, Larrañeta J, Ibáñez J, Montero G. Pilot demonstration of a web-based loading zone reservation system. In: The fourth international conference on city logistics. Malaysia: Institute of Systems Science Research; 2005. p. 401–15.
[6] McLeod F, Cherrett T. Loading bay booking and control for urban freight. International Journal of Logistics Research and Applications 2011;14:385–97.
[7] Consortium F. FREILOT Project. urban freight energy efficiency pilot, ⟨http://www.freilot.eu/⟩; 2011. Supported by the European Commission.
[8] Koulamas C. Maximizing the weighted number of on-time jobs in single machine scheduling with time windows. Mathematical and Computer Modelling 1997;25:57–62.

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010

[9] Koulamas C. Single-machine scheduling with time windows and earliness/tardiness penalties. European Journal of Operational Research 1996;91:190–202.
[10] Yin Y, Cheng T, Hsu C, Wu C. Single-machine batch delivery scheduling with assignable common due window. Omega 2013;41:216–25.
[11] Pan Q, Ruiz R. An estimation of distribution algorithm for lot-streaming flow shop problems with setup times. Omega 2012;40:166–80.
[12] Yenisey MM, Yagmahan B. Multi-objective permutation flow shop scheduling problem: literature review, classification and current trends. Omega 2014;45:119–35.
[13] Moccia L. New Optimization models and algorithms for the management of maritime container terminals [Ph.D. thesis]. Università della Calabria; 2004.
[14] Bolat A. Models and a genetic algorithm for static aircraft-gate assignment problem. The Journal of the Operational Research Society 2001;52:1107–20.
[15] Caprara A, Galli L, Toth P. Solution of the train platforming problem. Transportation Science 2011;45:246–57.
[16] Pinedo M. Scheduling: theory, algorithms, and systems. USA: Springer; 2005.
[17] Park Y, Kim K. A scheduling method for berth and quay cranes. OR Spectrum 2003;25:1–23.
[18] Golias M, Boile M, Theofanis S. The berth allocation problem: a formulation reflecting time window service deadlines. In: The 48th transportation research forum annual meeting, USA.
[19] Narciso ME, Piera MA. Robust gate assignment procedures from an airport management perspective. Omega 2015;50:82–95.
[20] Toth P, Vigo D. The vehicle routing problem. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics; 2002.
[21] Kubiak W. Proportional optimization and fairness. US: Springer Science and Business Media; 2008.
[22] Andreatta G, Mason F. Properties of k-centra in a tree network. Networks 1985;15:21–5.
[23] Garfinkel R, Fernández E, Lowe T. The k-centrum shortest path problem. TOP 2006;14:279–92.
[24] Slater P. Centers to centroids in a graph. Journal of Graph Theory 1978;2:209–22.
[25] Tamir A. The k-centrum multi-facility location problem. Discrete Applied Mathematics 2001;109:293–307.
[26] PROINTEC. Estudi metodològic i desenvolupament de projectes sobre propostes de millora de la distribució urbana i de les operacions de càrrega i descàrrega per a distribució de mercaderies a Barcelona. Methodological study and development of projects about improvements in urban distribution and loading/unloading operations, Barcelona Municipality; 1997.

Please cite this article as: Roca-Riu M, et al. Parking slot assignment for urban distribution: Models and formulations. Omega (2015), http://dx.doi.org/10.1016/j.omega.2015.04.010