Operations Research Letters 45 (2017) 334–341

Contents lists available at ScienceDirect

Operations Research Letters

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

Pickup and delivery problem with time windows: A new compact two-index formulation

Maria Gabriela S. Furtado, Pedro Munari, Reinaldo Morabito *

Federal University of São Carlos, Rodovia Washington Luís, Km 235 - São Carlos - São Paulo - CEP 13565-905, Brazil

ARTICLE INFO

Article history:
Received 21 July 2015
Received in revised form
29 April 2017
Accepted 29 April 2017
Available online 8 May 2017

Keywords:
Pickup and delivery
Time windows
Two-index formulation
Precedence constraints
Compact formulation

ABSTRACT

We propose a formulation for the pickup and delivery problem with time windows, based on a novel modeling strategy that allows the assignment of vehicles to routes explicitly in two-index flow formulations. It leads to an effective compact formulation that can benefit OR practitioners interested in solving the problem by general-purpose optimization software. Computational experiments indicate that the proposed formulation has interesting features and best overall performance in relation to other compact formulations.

1. Introduction

Consider a set of point-to-point transportation requests defined by customer demands that must be collected from pickup locations and delivered to paired delivery locations. A fleet of vehicles with limited capacity is available in a depot to service the requests. The pickup and delivery problem with time windows (PDPTW) involves designing minimum-cost routes for these vehicles to satisfy all the requests. The routes must respect the pairing and precedence relations of pickup and delivery points, in addition to the time windows imposed by them.

The problem arises in several real-life contexts, such as freight transportation, maritime shipping, urban courier services and door-to-door passenger transportation [2]. It is a generalization of the classical vehicle routing problem (VRP), and hence, it is also NP-hard [8]. Several formulations and solution methods are proposed for the PDPTW [15,5,14,3,1]. For comprehensive reviews, see [16,6,5,11].

The PDPTW can be modeled using a classical three-index vehicle flow formulation, which has the advantage of being compact *i.e.*, the number of variables and constraints is polynomial in terms of the number of customer requests. This allows straightforward coding and solving of the model using general-purpose optimization software. However, this formulation suffers from symmetry and a weak linear relaxation; moreover, the number of variables can become considerably large as the numbers of requests and vehicles increase. These drawbacks negatively affect the performance of optimization software, especially when solving large-scale instances. In [10], the authors propose a different compact formulation using two-index variables. The formulation is based on modifying the graph representation of the PDPTW so that the problem becomes finding a Hamiltonian tour that satisfies the pickup and delivery relations. To our knowledge, this is the only compact two-index model for the PDPTW so far. The authors also propose a branch-and-cut method with specific valid inequalities to solve the problem; the results indicate that only small to moderate-sized instances could be solved using this approach.

In this paper, we propose a new two-index formulation for the PDPTW in which the numbers of variables and constraints are polynomial with respect to the number of requests. This compact model is obtained by extending the classical two-index formulation of the vehicle routing problem with time windows (VRPTW) [7]. New continuous variables and constraints are created to ensure the pairing and precedence relations of pickup and delivery locations. We rely on a novel modeling strategy that explicitly assigns vehicles to routes in the two-index flow formulation. To the best of our knowledge, this strategy has never been used in the literature. We believe that the formulation of other variants of vehicle routing problems can also benefit from this idea.

We have run a set of computational experiments to verify the advantages of the proposed two-index model over other compact

* Corresponding author.
E-mail addresses: gabisfurtado@gmail.com (M.G.S. Furtado), munari@dep.ufscar.br (P. Munari), morabito@ufscar.br (R. Morabito).

http://dx.doi.org/10.1016/j.orl.2017.04.013
0167-6377/

formulations, namely, the classical three-index model and the two-index model proposed in [10]. The results using well-known PDPTW instances from the literature indicate that the proposed model provides better results than the others. We believe that a compact formulation that works reasonably well in practice can benefit OR practitioners interested in solving the PDPTW, as it allows the convenient use of general-purpose optimization software as a black-box.

In the literature, exponential-size formulations have also been proposed for the PDPTW. Although they allow the solving of larger instances, these formulations require developing and implementing sophisticated solution methods, which require time and computer programming skills. Ropke et al. [14] propose two-index formulations in which the number of constraints is exponential in terms of the number of customer requests. To solve the formulations, the authors develop a branch-and-cut method based on specialized valid inequalities. Ropke and Cordeau [13] and Baldacci et al. [1] propose set partitioning formulations for the PDPTW, which are defined by an exponential number of variables—one for each feasible route in the problem. Hence, column generation, valid inequalities and branch-and-price methods are proposed by the authors to solve the formulations. Comparing these exponential-size formulations against the compact formulation that we propose in this paper would require designing and implementing branch-and-cut and branch-and-price methods. Moreover, the comparisons would be biased by computational implementation details regarding valid inequalities, pricing subproblems and branching strategies, among others. Hence, we believe that such comparison is beyond the scope of this paper, as our aim is to present and analyze the benefits of a new compact formulation for the PDPTW.

The structure of this paper is as follows. In Section 2, we introduce the mathematical notation and briefly review the two compact formulations presented in the literature: the three-index model and the model proposed in [10]. In Section 3, we describe the new compact formulation with two-index variables for the PDPTW. The computational experiments are presented in detail in Section 4, followed by concluding remarks in Section 5.

2. Compact formulations of the PDPTW

In this section, we state the problem and introduce the mathematical notation that will be used in the remainder of the paper. Then, we briefly describe the classical three-index formulation of the PDPTW, followed by the two-index formulation proposed in [10], which we refer to here as the LD model.

2.1. Problem statement and notation

Let n be the total number of customer requests. The PDPTW can be represented by a complete graph $G(N, A)$ defined by node set N and arc set A, where $N = \{0, 1, \ldots, 2n+1\}$. Nodes 0 and $2n+1$ represent the origin and the final depots, respectively, which can have the same location. Each node $i \in N$ has a demand q_i and a non-negative service time d_i where $q_0 = q_{2n+1} = 0$ and $d_0 = d_{2n+1} = 0$. The subsets $P = \{1, \ldots, n\} \subset N$ and $D = \{n+1, \ldots, 2n\} \subset N$ are the pickup and delivery node sets, respectively. Each pickup request $i \in P$ is associated with a delivery node $n+i \in D$, such that $q_i \geq 0$ and $q_{n+i} = -q_i$. A time window $[e_i, l_i]$ is imposed for each node $i \in N$, such that e_i and l_i represent the earliest and latest time at which service is allowed to start at node i, respectively. The time windows of depot nodes 0 and $2n+1$ define the earliest and the latest time in which each vehicle must leave and return to the depot. Furthermore, a routing cost c_{ij} and a travel time t_{ij} are associated with each arc $(i, j) \in A$. We assume that the travel time t_{ij} includes the service time d_i at node i and that the triangle inequality holds for routing costs and travel times. Also, we consider a homogeneous fleet of vehicles such that $K = \{1, 2, \ldots, m\}$ is the set of identical vehicles with capacity Cap.

The problem consists in defining minimal cost routes to visit exactly once the pickup and delivery nodes of each request. Costs are typically defined equal to travel times (without service times) with an additional penalty for the number of routes (see Section 4). Each route must start at the initial depot node 0 and finish at the final depot node $2n+1$; satisfy the demand and time windows of the visited pickup and delivery nodes; and never exceed the vehicle capacity. Nodes can be visited at any order as long as the same route visits both the pickup and delivery nodes of a request (pairing relation) and the pickup node is visited earlier than the delivery node (precedence relation). The visit to a delivery node does not have to be immediately after the visit to its corresponding pickup node.

2.2. Three-index formulation of the PDPTW

In the classical three-index formulation of the PDPTW [4,13], a binary variable x_{ijk} is defined for each arc $(i, j) \in A$ and each vehicle $k \in K$, such that, $x_{ijk} = 1$ if and only if vehicle k visits node i and then travels directly to node j. Non-negative continuous variables Q_{ik} and B_{ik} respectively indicate the load of vehicle k after visiting node i and the time that the vehicle k starts servicing node i, for each $i \in N$ and each $k \in K$.

The PDPTW formulation using the defined variables is based on the standard three-index vehicle flow formulation of the VRPTW [9,7]. To adapt the VRPTW model to the PDPTW case, we add the two following set of constraints: $\sum_{j \in N} x_{ijk} - \sum_{j \in N} x_{n+i,j,k} = 0$ and $B_{n+i,k} \geq B_{ik} + t_{i,n+i}$, $\forall i \in P, k \in K$. The first set guarantees that the same vehicle visits both the pickup and delivery nodes of the request (pairing) and the second set of constraints ensures that the pickup node is visited before the delivery node (precedence). The resulting three-index formulation of the PDPTW is detailed in Appendix A.

2.3. Two-index formulation of the PDPTW (LD model)

The two-index formulation proposed by Lu and Dessouky [10] is based on an extended graph representation of the PDPTW so that the problem becomes finding a minimal cost Hamiltonian tour in the graph. Recall that $m = |K|$ is the number of vehicles in the fleet. The changes in the original graph representation consist of creating $m+1$ artificial nodes, which are used as departure and return depots for each of the vehicles. Let P and D be the subsets of pickup and delivery nodes as defined in Section 2.1. Now, let $N_0 = \{2n+1, 2n+2, \ldots, 2n+m+1\}$ be the set of $m+1$ artificial depots such that each vehicle $k \in K$ must depart from node $2n+k$ and return to node $2n+k+1$. For instance, the first vehicle must depart from node $2n+1$ and return to node $2n+2$ at the end of its route, while the last vehicle (m) must depart from node $2n+m$ and return to node $2n+m+1$. Notice that in this formulation the depot node $2n+1$ is now the initial depot of the first vehicle. Since all initial depot nodes belong to N_0, the node 0 is not included in this formulation.

Following [10], we represent the problem by creating an extended graph $\tilde{G}(\tilde{N}, \tilde{A})$, with the set of nodes $\tilde{N} = P \cup D \cup N_0$. The set of arcs $\tilde{A}$ consists of all arcs that connect: (i) all nodes in $P \cup D$; (ii) the depot nodes in N_0 as a cycle, i.e., $(2n+1, 2n+2)$, $(2n+2, 2n+3)$, $\ldots$, $(2n+m, 2n+m+1)$, $(2n+m+1, 2n+1)$; and (iii) any node in D to any other node in $N_0 \setminus \{2n+1\}$ and any node in $N_0 \setminus \{2n+m+1\}$ to any other in P. We consider N_0 without $\{2n+1\}$ in the first set of arcs, as $2n+1$ is the initial depot of the first vehicle and, hence, there can be no visit to any node before this depot node. Similarly, there can be no visit after node $2n+m+1$, as it is the final depot for vehicle m. To be consistent with the new

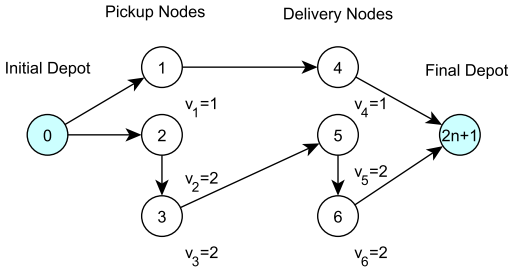

Fig. 1. Illustration of the values of v_i in an example with two routes.

graph, the definition of the routing cost c_{ij} and the travel time t_{ij} must be extended to each arc (i, j) in $\tilde{A}$. These values are set to zero for any arc that connects two depot nodes in N_0 and set in the usual way for the remaining arcs.

The decision variables in the LD model are based on the arcs of the extended graph. For each $(i, j) \in \tilde{A}$ there is a binary variable x_{ij} that is equal to 1 if and only if a route visits node i and travels directly to node j. The pairing and precedence relations are ensured by using the binary variable b_{ij}, for each $(i, j) \in \tilde{A}$, such that $b_{ij} = 1$ if and only if node i is visited before node j in the Hamiltonian tour. As shown in [10], the integrality of these variables can be relaxed without loss of generality. To guarantee the time window requirements, the continuous variable T_i gives the time at which the service starts at node i, for each $i \in \tilde{N}$, similar to the variable B_{ik} of the three-index formulation. Using all these variables, we obtain the compact MIP formulation proposed in [10], which is presented in Appendix A.

3. A new compact formulation for the PDPTW

Let $G(N, A)$ be the complete graph that represents the PDPTW, as defined in Section 2.1. For each arc $(i, j) \in A$, let x_{ij} be a binary variable that is equal to 1 if and only if a vehicle travels directly from node i to node j. For each $i \in P \cup D$, let Q_i be the non-negative continuous variable that indicates the vehicle load after visiting node i. In addition, let B_i be the non-negative continuous variable that indicates the time that a vehicle starts servicing node i. These variables have a similar meaning as the variables B_{ik} and T_i that are defined in Section 2. They can be used to satisfy the precedence relations of paired pickup and delivery nodes by using the constraints $B_{n+i} \geq B_i + t_{i,n+i}, \forall i \in P$.

To guarantee the pairing relations, we define additional variables that are used to identify the routes by storing the index of the first node that is visited in the route. Hence, pickup and delivery nodes for the same request must have the same identifier (*i.e.*, the same stored index). Formally, let v_i be a continuous decision variable that is equal to the index of the first node in the route that visits node $i \in P \cup D$. For instance, if j is the first node visited in the route that visits i, then we have $v_i = j$. As a consequence, the pickup and delivery nodes of a given request i are visited in the same route if and only if $v_i = v_{n+i}$. Fig. 1 illustrates the values of the variables v_i using an example with three requests. Nodes 1, 2 and 3 are the pickup locations of delivery nodes 4, 5 and 6, respectively. Two routes are used to service the requests. The first route starts visiting node 1, and hence, $v_i = 1$ for all nodes i in this route ($v_1 = v_4 = 1$). Similarly, the second route starts visiting node 2, and hence, we have $v_i = 2$ for all nodes in the route ($v_2 = v_3 = v_5 = v_6 = 2$). Note that all routes share the same origin and destination depot; thus, there is no need to include variables v_0 and v_{2n+1} in the formulation.

As mentioned above, the new compact formulation is based on the standard two-index formulation of the VRPTW. To ensure that paired pickup and delivery nodes will be in the same route, we add the constraints: $v_i = v_{n+i}, \forall i \in P$. In addition, we must state constraints that guarantee the consistency of the variables v_i in the model. To ensure that the route identifier is taken as the index of the first visited node, we use the following two set of constraints: $v_j \geq j \cdot x_{0j}$ and $v_j \leq j \cdot x_{0j} - n\left(x_{0j} - 1\right), \forall j \in P \cup D$. If node $j \in P \cup D$ is the first node of a route, we have $x_{0j} = 1$ and thus the two corresponding constraints together enforce that $v_j = j$. On the other hand, if j is not the first node of any route, then $x_{0j} = 0$ and it turns off the corresponding constraints (as together they impose the trivial bounds $0 \leq v_j \leq n$). We also must ensure that the index of the first node is forwarded to the next nodes in the route. We can do so by imposing the following two sets of constraints: $v_j \geq v_i + n\left(x_{ij} - 1\right)$ and $v_j \leq v_i + n\left(1 - x_{ij}\right), \forall i, j \in P \cup D$. This way, if there is a route that visits a node i and travels directly to a node j, then $x_{ij} = 1$ and the corresponding constraints imply that $v_j = v_i$. Otherwise, we have $x_{ij} = 0$, which turns off the corresponding constraints. Note that the variables v_i are implicitly integer due to these sets of constraints.

Using the variables and constraints defined above, we state the new compact two-index formulation for the PDPTW as the following MIP model:

$$\min \quad \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{i \in N} x_{ij} = 1 \qquad \forall j \in P \cup D \tag{2}$$

$$\sum_{j \in N} x_{ij} = 1 \qquad \forall i \in P \cup D \tag{3}$$

$$B_j \geq B_i + t_{ij} - M\left(1 - x_{ij}\right) \qquad \forall i \in N; j \in N \tag{4}$$

$$Q_j \geq Q_i + q_j - M\left(1 - x_{ij}\right) \qquad \forall i \in N; j \in N \tag{5}$$

$$e_i \leq B_i \leq l_i \qquad \forall i \in N \tag{6}$$

$$\max\{0, q_i\} \leq Q_i \leq \min\{Cap, Cap + q_i\} \qquad \forall i \in N \tag{7}$$

$$B_{n+i} \geq B_i + t_{i,i+n} \qquad \forall i \in P \tag{8}$$

$$v_{n+i} = v_i \qquad \forall i \in P \tag{9}$$

$$v_j \geq j \cdot x_{0j} \qquad \forall j \in P \cup D \tag{10}$$

$$v_j \leq j \cdot x_{0j} - n\left(x_{0j} - 1\right) \qquad \forall j \in P \cup D \tag{11}$$

$$v_j \geq v_i + n\left(x_{ij} - 1\right) \qquad \forall i, j \in P \cup D \tag{12}$$

$$v_j \leq v_i + n\left(1 - x_{ij}\right) \qquad \forall i, j \in P \cup D \tag{13}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in N; j \in N. \tag{14}$$

The objective function (1) minimizes the total routing costs as in the other compact formulations addressed in Section 2. Constraints (2) and (3) guarantee that each node is visited exactly once. The consistency of time variables and load variables is ensured by constraints (4) and (5), respectively, where the constant M is defined as a sufficiently large number. The time windows and the vehicle capacity are guaranteed by constraints (6) and (7), respectively. Moreover, constraints (4) and (6) guarantee that there are no subtours in the solutions. Constraints (8)–(13) guarantee the precedence and pairing relations as described above. Finally, constraints (14) impose integrality for the two-index variables.

4. Computational experiments

In this section, we describe a set of computational experiments to analyze the performance of the compact two-index formulation

proposed in Section 3. The purpose is to compare this formulation against the three-index model and the LD model addressed in Section 2. The three formulations were implemented using the IBM OPL modeling language and solved with the IBM CPLEX Optimization Studio version 12.5.1 and with the optimization software Gurobi version 6.0.4.

In the implementation of the two-index formulation, we have included the following trivial variable fixations, to speed up the running times: $x_{ii} = 0, x_{i0} = 0$ and $x_{n+1,i} = 0, \forall i \in P \cup D$; $x_{i,n+1} = 0$ and $x_{i+n,i} = 0, \forall i \in P$; $x_{0,i} = 0, \forall i \in D$; $x_{ij} = 0, \forall i, j \in P \cup D$ such that $e_i + t_{ij} > l_j$; $x_{ij} = 0$, $x_{i,j+n} = 0$ and $x_{i+n,j+n} = 0$, $\forall i, j \in P$ such that $q_i + q_j > Cap$. Equivalent fixations were added to the implementations of the other formulations. They are imposed as variable bounds and hence are useful to reduce the problem size on preprocessing. Results obtained without these fixations were rather inferior for all formulations, so we use them in all experiments reported ahead in this section. Also, we set the Big-M values of the timing and load constraints (4) and (5) respectively as $l_i - e_j$ and $Cap + q_i$. Again, we set the Big-M values of the other models in a similar way.

The computer used to run all experiments was a PC Core i7 2.10 GHz (eight cores), 8 GB of RAM with a Windows 7 Home Premium operating system. For all experiments, we used a limit of two hours of CPU time for each instance. Also, we kept the standard parameters settings of the OPL user interface, including multi-thread execution and MIP optimality tolerance equal to 10^{-4}.

The algorithms were tested using the problem instances proposed in [13] for the PDPTW. There are four classes of instances (*AA*, *BB*, *CC* and *DD*) with different vehicle capacity *Cap* and time windows with width *W*. Class *AA* has $Cap = 15$ and $W = 60$; class *BB* has $Cap = 20$ and $W = 60$; class *CC* has $Cap = 15$ and $W = 120$; and class *DD* has $Cap = 20$ and $W = 120$. Each class consists of 10 instances and the number of pairs of pickup and delivery nodes of these instances are multiples of five. The smallest instance of a class has 30 pairs and the largest instance has 75 pairs. The class name and the number of pairs of pickup and delivery nodes determine the instance name. For example, *AA*30 is the instance in class *AA* with $|P| = |D| = 30$. The instances consider homogeneous fleet and a single depot located at the middle of a $[0, 50] \times [0, 50]$ square. The coordinates of the pickup and delivery nodes are randomly distributed in the same square. For further details regarding these instances, please see [13].

In addition to the instances proposed in [13], we have created five new small instances for each class, with the number of requests ranging from 5 to 25. They were obtained by keeping only the first *r* requests of the instance with 30 requests of each class, with $r = 5, 10, \ldots, 25$. For example, for class *AA*, we have created instance *AA*5 by keeping only the first five requests of *AA*30, *AA*10 has only the 10 first requests, and so on.

Finally, the objective functions of the models are implemented as in the other papers addressing the PDPTW. Namely, arc costs are equal to travel costs (without service times) and we add the penalty value 10^4 to the cost of each arc that leaves the initial depot (c_{0i}) to minimize the number of vehicles used in the optimal solution.

Table 1 gives the results of the experiments using the LD model, the three-index model and the proposed two-index model, all solved by the optimization software CPLEX with its default parameter settings. The first column in Table 1 identifies the name of the instance. For each model, we provide the upper bound (UB) followed by the time given in seconds (Time), the relative gap provided by CPLEX (*Gap*) and the relative gap with respect to the best known solution for the instance (Gap_{Best}), both given as a percentage. The relative gap is calculated by $Gap = 100 \times \frac{UB-LB}{UB}$ in which *UB* is the best upper bound and *LB* is the best lower bound found by CPLEX. The Gap_{Best} is calculated by $Gap_{Best} = 100 \times \frac{UB-z_{Best}}{UB}$ in which *UB* is the best upper bound provided by CPLEX when the time limit is reached, and z_{Best} is the value of the best known solution for the instance, as provided in [13,1]. The symbol $*$ indicates that CPLEX found the optimal solution for the corresponding instance using the compact two-index formulation proposed in this paper. When an instance could not be solved to optimality in two hours, then we report only the upper bound and the corresponding *Gap* and Gap_{Best}. In these cases, the column Time has the symbol –. The last line in the table summarizes the results for *Gap* and Gap_{Best} by showing the arithmetic mean over all instances. For the instances for which no gap is available (N/A), as no feasible solution was found within the imposed time limit, we consider $gap = 100\%$ to calculate the average gap of all instances.

In Table 1, we observe that the LD model solves to optimality only 10 instances (out of 60 instances) within the time limit, which are the smallest instances in the set. The three-index model solves only 15 instances, while the proposed two-index formulation solves 26 instances within a reasonable running time. Moreover, the proposed model finds the optimal solutions of six other instances (*AA*50, *AA*55, *BB*45, *BB*60, *CC*30 and *CC*35), although it could not prove optimality for them (based on the optimal solutions provided in [13,1]). The same happens with the three-index model with different four instances (*BB*30, *CC*20, *CC*25 and *DD*15) and with the LD model for one instance only (*BB*15). Furthermore, the average gap is equal to 77.7% for the LD model, 60.4% for the three-index model and 31.1% for the two-index model; the average Gap_{Best} for these three models is 77.2%, 38.8% and 6.9%, respectively. Considering all instances, the proposed two-index formulation is never worse than the two others, in terms of both time and solution quality. It was the only one to find feasible solutions to all instances. In addition, this formulation was able to obtain optimal solutions to larger instances compared to the other formulations.

To verify the influence of the parameter settings of CPLEX to the performance of the formulations, we have run additional experiments using different parameter choices. The three models were solved by CPLEX, but with its cuts and primal heuristics turned off. The cuts automatically generated by CPLEX are of the following type: clique, cover, disjunctive, flow cover, flow path, Gomory, GUB cover, implied bound, mixed integer rounding and zero-half. We have also checked the behavior of the three models in another general-purpose optimization solver. The solver Gurobi has been used to solve all instances, in its default parameter settings. The same time limit of two hours was considered in these experiments. The results are summarized in Table 2. The first row in the table corresponds to the same results presented in Table 1, which uses the default parameter settings of CPLEX. The next two rows give the results of the experiments using CPLEX without cuts (row 2) and without primal heuristics (row 3). The last row presents the results obtained with the solver Gurobi. For each model type, the table has four columns: *Opt* gives the total number of instances for which the solver proved optimality; *No sol* gives the total number of instances for which the solver could not find a feasible solution within the time limit; and *Gap* and Gap_{Best} show the average over the gaps of all instances, where the gaps are computed as mentioned above and given as percentages.

The results in Table 2 indicate that the proposed two-index formulation still obtains the best overall results even when turning off the cuts and primal heuristics of CPLEX, compared to the performance of the other formulations (using or not cuts and heuristics). Turning off cuts in CPLEX had little impact for the LD model and the three-index model, while we observe a performance reduction for the proposed formulation. This is probably because the results for the former two models using CPLEX with its default parameters were already very inferior than the results obtained by the proposed formulation. Turning off heuristics had a more negative impact than turning off cuts, especially to the three-index model and the proposed two-index model. The results using

Table 1
Comparison of the compact two-index and three-index formulations.

Instance	LD model				Three-index model				Two-index model			
	UB	Time	*Gap*	Gap_{Best}	UB	Time	*Gap*	Gap_{Best}	UB	Time	*Gap*	Gap_{Best}
*AA*5∗	10 184.8	0.41	0	0	10 184.8	0.40	0	0	10 184.8	0.23	0	0
*AA*10∗	10 383.6	8.87	0	0	10 383.6	1.42	0	0	10 383.6	0.34	0	0
*AA*15∗	20 542.4	–	0	0	20 542.4	8.18	0	0	20 542.4	0.53	0	0
*AA*20∗	20 769.3	–	0.46	0.05	20 759.4	41.0	0	0	20 759.4	0.66	0	0
*AA*25∗	N/A	N/A	N/A	N/A	21 055.3	3388.9	0	0	21 055.3	2.00	0	0
*AA*30∗	N/A	N/A	N/A	N/A	31 122.8	–	31.1	0.01	31 119.1	22.2	0	0
*AA*35∗	N/A	N/A	N/A	N/A	31 307.2	–	31.1	0.02	31 299.8	532.2	0	0
*AA*40∗	N/A	N/A	N/A	N/A	41 498.4	–	24.4	24.0	31 515.9	726.6	0	0
*AA*45	N/A	N/A	N/A	N/A	71 946.6	–	63.1	55.8	41 644.5	–	24.0	23.7
*AA*50∗	N/A	N/A	N/A	N/A	92 235.7	–	76.5	54.7	41 775.0	–	0.19	0
*AA*55∗	N/A	N/A	N/A	N/A	102 561.3	–	78.8	59.1	41 907.8	–	0.2	0
*AA*60	N/A	N/A	N/A	N/A	102 916.1	–	78.6	59.0	52 060.2	–	28.68	19.1
*AA*65	N/A	N/A	N/A	N/A	102 943.7	–	78.6	58.9	52 181.2	–	38.2	19.0
*AA*70	N/A	N/A	N/A	N/A	153 351.5	–	92.1	72.3	52 356.0	–	19.3	18.9
*AA*75	N/A	N/A	N/A	N/A	173 583.9	–	92.9	69.8	52 486.1	–	19.3	0.04
*BB*5∗	10 339.7	0.44	0	0	10 339.7	0.18	0	0	10 339.7	0.17	0	0
*BB*10∗	20 512.5	222.2	0	0	20 512.5	0.95	0	0	20 512.5	0.36	0	0
*BB*15∗	20 667.6	–	0.07	0	20 667.6	2.89	0	0	20 667.6	0.45	0	0
*BB*20∗	20 842.3	–	0.39	0.24	20 791.3	7.27	0	0	20 791.3	0.64	0	0
*BB*25∗	N/A	N/A	N/A	N/A	20 988.0	56.3	0	0	20 988.0	2.33	0	0
*BB*30∗	N/A	N/A	N/A	N/A	31 086.3	–	31.9	0	31 086.3	1.96	0	0
*BB*35∗	N/A	N/A	N/A	N/A	31 281.7	–	31.9	0.001	31 281.2	28.6	0	0
*BB*40∗	N/A	N/A	N/A	N/A	41 473.8	–	48.3	24.1	31 493.4	87.1	0	0
*BB*45∗	N/A	N/A	N/A	N/A	61 812.5	–	81.3	32.8	41 555.1	–	24.1	0
*BB*50	N/A	N/A	N/A	N/A	92 212.8	–	87.3	54.8	41 705.9	–	0.1	0.01
*BB*55	N/A	N/A	N/A	N/A	122 604.2	–	90.4	65.8	51 870.9	–	19.5	19.2
*BB*60∗	N/A	N/A	N/A	N/A	123 110.9	–	82.1	49.3	62 420.3	–	28.3	0
*BB*65	N/A	N/A	N/A	N/A	123 415.6	–	81.9	49.2	62 648.1	–	24.1	0.01
*BB*70	N/A	N/A	N/A	N/A	163 883.8	–	86.2	61.6	72 843.9	–	37.6	13.6
*BB*75	N/A	N/A	N/A	N/A	174 073.8	–	87.0	63.7	72 963.3	–	40.1	13.5
*CC*5∗	10 212.5	0.55	0	0	10 212.5	0.27	0	0	10 212.5	0.17	0	0
*CC*10∗	20 385.5	18.1	0	0	20 385.5	1.03	0	0	20 385.5	0.45	0	0
*CC*15∗	20 571.2	5121.6	0	0	20 571.2	13.1	0	0	20 571.2	2.77	0	0
*CC*20∗	30 813.6	–	65.0	32.6	20 764.4	–	47.7	0	20 764.4	25.9	0	0
*CC*25∗	N/A	N/A	N/A	N/A	20 944.4	–	47.6	0	20 944.4	136.6	0	0
*CC*30∗	N/A	N/A	N/A	N/A	31 137.2	–	96.2	0.16	31 087.7	–	31.9	0
*CC*35∗	N/A	N/A	N/A	N/A	41 381.8	–	97.1	24.5	31 230.6	–	31.8	0
*CC*40	N/A	N/A	N/A	N/A	81 867.3	–	98.4	61.7	31 364.5	–	63.5	0.02
*CC*45	N/A	N/A	N/A	N/A	102 164.9	–	98.6	69.1	41 505.8	–	85.7	24.1
*CC*50	N/A	N/A	N/A	N/A	132 651.1	–	98.8	68.6	41 694.8	–	95.4	0.02
*CC*55	N/A	N/A	N/A	N/A	102 559.7	–	98.5	59.2	51 840.6	–	96.7	19.3
*CC*60	N/A	N/A	N/A	N/A	223 287.1	–	99.2	81.2	51 994.8	–	96.4	19.2
*CC*65	N/A	N/A	N/A	N/A	223 596.0	–	99.2	81.1	52 174.5	–	96.4	19.2
*CC*70	N/A	N/A	N/A	N/A	574 371.6	–	99.9	90.9	62 317.0	–	96.9	16.2
*CC*75	N/A	N/A	N/A	N/A	634 656.6	–	99.9	91.7	62 571.1	–	96.7	16.3
*DD*5∗	10 324.8	0.19	0	0	10 324.8	0.23	0	0	10 324.8	0.12	0	0
*DD*10∗	20 567.8	4177.5	0	0	20 567.8	6.53	0	0	20 567.8	0.91	0	0
*DD*15∗	N/A	N/A	N/A	N/A	20 673.4	–	48.2	0	20 673.4	18.7	0	0
*DD*20∗	N/A	N/A	N/A	N/A	20 792.4	–	48.7	0.08	20 776.2	47.8	0	0
*DD*25∗	N/A	N/A	N/A	N/A	41 244.7	–	98.0	49.1	21 004.9	6158.2	0	0
*DD*30	N/A	N/A	N/A	N/A	51 422.1	–	98.3	58.9	21 156.7	–	0.3	0.11
*DD*35	N/A	N/A	N/A	N/A	91 876.3	–	99.0	66.0	31 231.2	–	64.2	0.06
*DD*40	N/A	N/A	N/A	N/A	82 009.7	–	98.8	61.8	31 379.5	–	64.2	0.09
*DD*45	N/A	N/A	N/A	N/A	112 225.2	–	99.0	71.9	31 509.9	–	64.0	0.08
*DD*50	N/A	N/A	N/A	N/A	142 710.4	–	99.2	77.8	41 622.8	–	96.6	24.1
*DD*55	N/A	N/A	N/A	N/A	193 026.3	–	99.4	83.5	41 804.8	–	96.6	24.1
*DD*60	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	52 053.5	–	97.0	38.4
*DD*65	N/A	N/A	N/A	N/A	183 479.9	–	99.2	77.0	52 179.7	–	96.6	19.3
*DD*70	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	62 323.5	–	97.1	32.2
*DD*75	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	62 582.1	–	96.9	32.3
Average	–	–	77.7%	77.2%	–	–	60.4%	38.8%	–	–	31.1%	6.9%

Table 2
Summary of experiments using the solver CPLEX with different parameter choices and using the solver Gurobi.

Solver/parameter settings	LD model				Three-index model				Two-index model			
	Opt	No Sol	*Gap*	Gap_{Best}	Opt	No Sol	*Gap*	Gap_{Best}	Opt	No Sol	*Gap*	Gap_{Best}
CPLEX (default)	10	46	77.7	77.2	15	3	60.4	38.8	26	0	31.1	6.9
CPLEX without cuts	9	46	77.4	76.6	16	5	58.2	38.5	25	13	36.9	25.4
CPLEX without heuristics	9	47	79.1	78.3	14	29	66.1	55.2	25	16	41.6	33.9
Gurobi (default)	11	43	74.9	72.7	14	27	63.5	50.1	25	0	30.7	6.0

Table 3
Number of variables and constraints in the three formulations in relation to the number of requests in the instance.

Requests	LD model		Three-index model		Two-index model		Two-index/LD model		Two-index/Three-index	
	Var.	Constr.	Var.	Constr.	Var.	Constr.	Var.	Constr.	Var.	Constr.
5	523	6 032	841	1 630	179	582	0.342	0.096	0.213	0.357
10	1 943	50 712	5 281	10 360	549	1 952	0.283	0.038	0.104	0.188
15	4 263	174 542	16 321	32 190	1 119	4 122	0.262	0.024	0.069	0.128
20	7 483	418 022	36 961	73 120	1 889	7 092	0.252	0.017	0.051	0.097
25	11 603	821 652	70 201	139 150	2 859	10 862	0.246	0.013	0.041	0.078
30	16 623	1 425 932	119 041	236 280	4 029	15 432	0.242	0.011	0.034	0.065
35	22 543	2 271 362	186 481	370 510	5 399	20 802	0.239	0.009	0.029	0.056
40	29 363	3 398 442	275 521	547 840	6 969	26 972	0.237	0.008	0.025	0.049
45	37 083	4 847 672	389 161	774 270	8 739	33 942	0.236	0.007	0.022	0.044
50	45 703	6 659 552	530 401	1 055 800	10 709	41 712	0.234	0.006	0.020	0.040
55	55 223	8 874 582	702 241	1 398 430	12 879	50 282	0.233	0.006	0.018	0.036
60	65 643	11 533 262	907 681	1 808 160	15 249	59 652	0.232	0.005	0.017	0.033
65	76 963	14 676 092	1 149 721	2 290 990	17 819	69 822	0.232	0.005	0.015	0.030
70	89 183	18 343 572	1 431 361	2 852 920	20 589	80 792	0.231	0.004	0.014	0.028
75	102 303	22 576 202	1 755 601	3 499 950	23 559	92 562	0.230	0.004	0.013	0.026

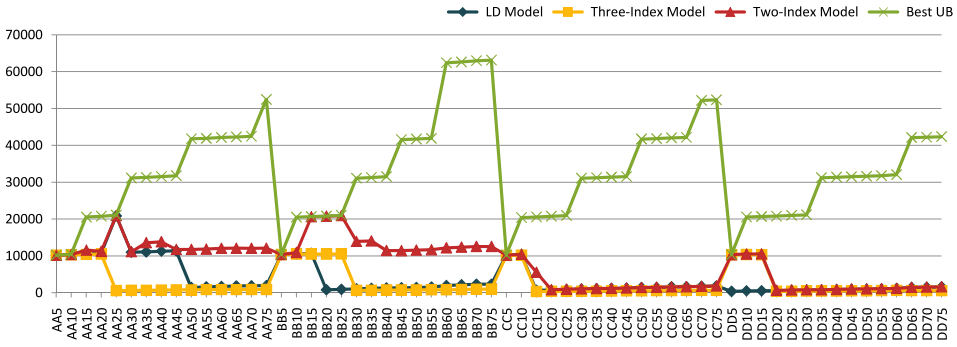

Fig. 2. Lower bounds provided by the linear relaxations of the three models and best known upper bounds.

the solver Gurobi confirm those obtained by CPLEX with default parameter settings and indicate that the proposed two-index formulation has the best overall performance.

We have run additional experiments to investigate why the performance of the proposed two-index formulation is better than the other two formulations. Table 3 shows the number of variables and constraints for each problem instance and each formulation (prior to CPLEX preprocessing), in relation to the number of requests in the instance. Recall that the classes of instances differ only by vehicle capacity (*Cap*) and time windows width (*W*). Hence, for a given formulation, instances with the same number of requests lead to the same number of variables and constraints. For example, the row for 5 requests refers to instances *AA*5, *BB*5, *CC*5 and *DD*5, as they all result in the same number of variables and constraints. The last four columns in the table give the ratio of these values for the two-index formulation in relation to the values for each of the other two formulations. Table 4 shows the optimal value of the linear relaxation (Relax.) for the three formulations, the total time in seconds (Time) to solve the relaxations and the ratio between the optimal values of the two-index formulation and each of the other two formulations. Again, the time limit for these tests was set as 2 h of CPU time.

As we can see in Table 3, the two-index formulation has the smallest number of variables and constraints among the three formulations for all instances. For most of these instances, the number of variables in the two-index model corresponds to less than 25% of the number of variables in the LD model and less than 5% of the number of variables in the three-index model. We believe this is one reason why the two-index formulation achieves better results than the three-index model and the LD model. Moreover, in the vast majority of instances, the linear relaxation of the two-index model results in optimal values that are greater than or equal to those obtained with the linear relaxations of the other models, as shown in Table 4. In some instances, the value of the linear relaxation of the two-index model is considerably larger than those of the other two models (e.g., *AA*25, *AA*50, *BB*20, among others). The main reason is the (trivial) variable fixation described in the beginning of Section 4. Although we use the same variable fixations in all models, they are stronger for the two-index formulation, due to the model structure. Additionally, it is worth noting that this linear relaxation was solved in just a few seconds for all instances, while the other formulations required more than one hour for some instances, or could not even finish solving the problem in two hours of CPU time (instances with the symbol – in Table 4). The average time to solve the linear relaxations was 2647.82 s for the LD model, 1240.52 s for the three-index model and only 0.11 s for the two-index model.

The results from Table 4 are summarized in Fig. 2, which presents the lower bounds provided by the linear relaxations of each model and the best known upper bounds, for all instances. As we can observe, the lower bounds provided by the proposed two-index model and the LD model are close and consistently better than those provided by the three-index model, for most instances.

5. Concluding remarks

The proposed formulation can be useful to solve small- to medium-sized instances to optimality in a reasonable amount of time using general-purpose optimization software. Feasible solutions with relatively good quality can be obtained for large-scale instances. Future work will focus on deriving effective valid inequalities for the proposed model and developing a customized branch-and-cut method to verify potential advantages over other formulations used with this type of method. Our research agenda also includes extending the model to address heterogeneous fleet and testing the model in practical situations with additional

Table 4
Objective values and solution times of the linear relaxations of the three models.

Instance	LD model		Three-index model		Two-index model		Two-index/LD model	Two-index/Three-index
	Relax.	Time	Relax.	Time	Relax.	Time	Relax.	Relax.
AA5	10 159.36	0.07	10 139.93	0.16	10 146.56	0.00	0.9987	1.0007
AA10	10 341.09	0.19	10 323.98	0.07	10 348.49	0.01	1.0007	1.0024
AA15	10 503.02	0.76	10 373.75	0.29	11 578.12	0.01	1.1024	1.1161
AA20	10 626.00	8.39	10 499.56	1.89	11 235.06	0.02	1.0573	1.0701
AA25	20 849.79	47.42	570.12	5.57	20 887.83	0.03	1.0018	36.6376
AA30	10 985.25	115.35	646.59	13.98	11 102.11	0.04	1.0106	17.1703
AA35	11 077.91	324.82	677.50	35.20	13 576.55	0.05	1.2256	20.0392
AA40	11 270.16	744.06	705.72	74.41	13 831.03	0.07	1.2272	19.5984
AA45	11 359.33	1442.62	772.47	179.88	11 724.31	0.10	1.0321	15.1777
AA50	1424.77	3162.20	776.96	370.46	11 737.39	0.12	8.2381	15.1068
AA55	1571.56	6279.16	834.00	698.27	11 865.36	0.13	7.5500	14.2270
AA60	1727.92	–	847.43	1195.84	12 039.69	0.17	6.9677	14.2073
AA65	1820.83	–	861.32	1825.27	12 099.62	0.19	6.6451	14.0478
AA70	1867.72	–	881.70	3621.36	12 055.32	0.29	6.4546	13.6729
AA75	1932.85	–	891.77	–	12 082.99	0.28	6.2514	13.5494
BB5	10 317.19	1.01	10 339.67	0.03	10 339.67	0.00	1.0022	1.0000
BB10	10 525.70	0.25	10 488.98	0.12	10 909.86	0.01	1.0365	1.0401
BB15	10 666.44	2.51	10 501.73	0.47	20 625.30	0.01	1.9337	1.9640
BB20	789.08	11.44	10 497.17	1.57	20 744.89	0.02	26.2898	1.9762
BB25	919.32	34.08	10 530.77	5.54	20 913.20	0.03	22.7485	1.9859
BB30	1021.69	129.04	607.52	12.55	13 924.16	0.04	13.6285	22.9198
BB35	1202.76	289.58	659.47	33.90	14 081.95	0.07	11.7081	21.3535
BB40	1286.52	923.95	749.52	80.53	11 440.20	0.11	8.8923	15.2634
BB45	1320.43	2678.50	738.51	189.85	11 430.92	0.16	8.6570	15.4783
BB50	1407.52	3804.93	773.50	392.53	11 547.55	0.13	8.2042	14.9289
BB55	1533.76	5566.00	816.95	863.25	11 657.87	0.16	7.6008	14.2700
BB60	1967.27	–	824.97	1133.90	12 182.44	0.17	6.1926	14.7671
BB65	2134.44	–	881.47	2107.01	12 372.44	0.23	5.7966	14.0361
BB70	2298.87	–	967.06	3791.16	12 562.80	0.25	5.4648	12.9908
BB75	2308.87	–	998.59	–	12 564.92	0.27	5.4420	12.5827
CC5	10 212.53	0.62	10 113.80	0.03	10 212.53	0.00	1.0000	1.0098
CC10	10 396.60	0.18	10 213.72	0.11	10 402.30	0.01	1.0005	1.0185
CC15	650.27	2.94	340.18	0.64	5524.90	0.01	8.4963	16.2411
CC20	727.79	12.82	361.14	2.63	826.14	0.02	1.1351	2.2876
CC25	869.68	40.10	402.98	8.76	956.00	0.03	1.0993	2.3723
CC30	982.23	141.33	403.78	17.88	1037.99	0.05	1.0568	2.5707
CC35	1105.24	296.91	438.94	52.61	1142.81	0.05	1.0340	2.6036
CC40	1146.68	925.83	476.63	129.74	1167.90	0.09	1.0185	2.4503
CC45	1242.95	1863.50	514.91	290.65	1263.17	0.10	1.0163	2.4532
CC50	1391.25	2821.56	519.43	569.12	1414.33	0.13	1.0166	2.7228
CC55	1445.25	5391.20	539.76	722.91	1459.29	0.16	1.0097	2.7036
CC60	1552.52	–	568.75	–	1569.31	0.19	1.0108	2.7592
CC65	1627.31	–	595.42	–	1633.90	0.21	1.0041	2.7441
CC70	1702.42	–	631.67	2150.17	1721.73	0.25	1.0113	2.7257
CC75	1805.82	–	651.51	3111.41	1838.50	0.34	1.0181	2.8219
DD5	382.12	0.33	10 287.35	0.01	10 324.77	0.00	27.0194	1.0036
DD10	489.07	0.25	10 342.69	0.07	10 511.15	0.01	21.4921	1.0163
DD15	530.21	1.26	10 321.39	0.35	10 557.55	0.01	19.9119	1.0229
DD20	573.53	1.76	363.79	2.16	635.59	0.02	1.1082	1.7471
DD25	693.75	17.93	396.66	8.42	754.65	0.03	1.0878	1.9025
DD30	768.39	72.16	446.91	27.04	802.28	0.04	1.0441	1.7952
DD35	797.81	96.62	443.94	68.02	818.89	0.06	1.0264	1.8446
DD40	870.62	232.01	479.96	135.35	904.67	0.09	1.0391	1.8849
DD45	965.89	439.68	497.43	306.61	1003.82	0.13	1.0393	2.0180
DD50	1010.37	2309.46	534.81	527.95	1038.04	0.14	1.0274	1.9410
DD55	1062.85	3434.27	563.42	1091.25	1080.60	0.18	1.0167	1.9179
DD60	1147.03	–	600.40	–	1169.47	0.22	1.0196	1.9478
DD65	1441.78	–	573.59	–	1465.60	0.21	1.0165	2.5551
DD70	1478.06	–	569.01	2126.77	1505.86	0.28	1.0188	2.6465
DD75	1582.45	–	581.06	3245.73	1599.87	0.32	1.0110	2.7534

constraints, such as real-life PDPTW instances derived from a Brazilian oil company [12].

Acknowledgments

The authors would like to thank the anonymous reviewer for her/his useful comments and valuable contribution to this paper. Also, they are thankful to FAPESP—São Paulo Research Foundation (Projects 2014/22542-2, 2014/00939-8 and 2010/10133-0) and CAPES (grant number 1256579) for financial support.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.orl.2017.04.013.

References

[1] R. Baldacci, E. Bertolini, A. Mingozzi, An exact algorithm for the pickup and delivery problem with time windows, Oper. Res. 59 (2) (2011) 414–426.
[2] M. Battarra, J.-F. Cordeau, M. Iori, Pickup-and-delivery problems for goods transportation, in: P. Toth, D. Vigo (Eds.), Vehicle Routing: Problems, Methods, and Applications, in: MOS/SIAM Ser Optim, 2014, pp. 161–191.

[3] N. Bianchessi, G. Righini, Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery, Comput. Oper. Res. 34 (2) (2007) 578–594.
[4] J.F. Cordeau, Branch and cut algorithm for the dial-a-ride problem, Oper. Res. 54 (3) (2006) 573–586.
[5] J.-F. Cordeau, G. Laporte, J.-Y. Potvin, M.W. Savelsbergh, Transportation on demand, Handbooks Oper. Res. Management Sci. 14 (2007) 429–466.
[6] G. Desaulniers, J. Desrosiers, A. Erdmann, M.M. Solomon, VRP with pickup and delivery, in: P. Toth, D. Vigo (Eds.), The Vehicle Routing Problem, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002, pp. 225–242.
[7] G. Desaulniers, O.B. Madsen, S. Ropke, The vehicle routing problem with time windows, in: P. Toth, D. Vigo (Eds.), Vehicle Routing: Problems, Methods, and Applications, in: MOS/SIAM Ser Optim, 2014, pp. 119–159.
[8] R.M. Garey, D.S. Johnson (Eds.), Computers and Intractability. A Guide to the Theory of NP-Completeness, Bell Laboratories, Murray Hill, NJ, 1979.
[9] B. Kallehauge, J. Larsen, O.B. Madsen, M.M. Solomon, Vehicle routing problem with time windows, in: G. Desaulniers, J. Desrosiers, M.M. Solomon (Eds.), Column Generation, Springer, US, 2005, pp. 67–98.
[10] Q. Lu, M. Dessouky, An exact algorithm for the multiple vehicle pickup and delivery problem, Transp. Sci. 38 (4) (2004) 503–514.
[11] S. Parragh, K. Doerner, R. Hartl, A survey on pickup and delivery problems: Part II: Transportation between pickup and delivery locations, J. Betriebswirtsch. 58 (2) (2008) 81–117.
[12] V.P. Rodrigues, R. Morabito, D. Yamashita, B.J. da Silva, P.C. Ribas, Ship routing with pickup and delivery for a maritime oil transportation system: MIP model and heuristics, Systems 4 (3) (2016) 31.
[13] S. Ropke, J.F. Cordeau, Branch and cut and price for the pickup and delivery problem with time windows, Transp. Sci. 43 (3) (2009) 267–286.
[14] S. Ropke, J.F. Cordeau, G. Laporte, Models and branch-and-cut algorithms for pickup and delivery problems with time windows, Networks 49 (4) (2007) 258–272.
[15] S. Ropke, D. Pisinger, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, Transp. Sci. 40 (4) (2006) 455–472.
[16] M.W.P. Savelsbergh, M. Sol, The general pickup and delivery problem, Transp. Sci. 29 (1) (1995) 17–29.