Optim Lett
DOI 10.1007/s11590-017-1138-5

ORIGINAL PAPER

Integer programming models and linearizations for the traveling car renter problem

Marco C. Goldbarg[1] · **Elizabeth F. G. Goldbarg**[1] · **Henrique P. L. Luna**[2] · **Matheus S. Menezes**[3] · **Lucas Corrales**[4]

Received: 18 July 2016 / Accepted: 28 March 2017

Abstract The traveling car renter problem (CaRS) is an extension of the classical traveling salesman problem (TSP) where different cars are available for use during the salesman's tour. In this study we present three integer programming formulations for CaRS, of which two have quadratic objective functions and the other has quadratic constraints. The first model with a quadratic objective function is grounded on the TSP interpreted as a special case of the quadratic assignment problem in which the assignment variables refer to visitation orders. The second model with a quadratic objective function is based on the Gavish and Grave's formulation for the TSP. The model with quadratic constraints is based on the Dantzig–Fulkerson–Johnson's formulation for the TSP. The formulations are linearized and implemented in two solvers. An experiment with 50 instances is reported.

Keywords Traveling car renter problem · Traveling salesman · Integer programming · Combinatorial optimization

1 Introduction

The car rental industry has been growing globally over the last few years and is still expected to grow at a compound annual growth rate of 5.6% on the next five years

✉ Elizabeth F. G. Goldbarg
beth@dimap.ufrn.br

1 Universidade Federal do Rio Grande do Norte, Natal, Brazil

2 Universidade Federal de Alagoas, Maceió, Brazil

3 Universidade Federal Rural do Semi-Árido, Mossoró, Brazil

4 Universidad Nacional del Centro de la Provincia de Buenos Aires, Tandil, Argentina

Published online: 03 April 2017

[22]. Part of this growth results from the expansion of the travel and tourism industry. Car rental also affects another markets such as the airport industry as pointed out by Czerny et al. [1] who showed that a one-dollar increase in the daily car rental price reduced significantly the demand at some US airports. Intense competition in the car rental industry has led companies to adopt information technologies which have impacted their results. Researches in this area have been driven under the viewpoint of the car rental industry. For example, recent studies investigated vehicle reservation and assignment [11,14,16,17], profit management [13] and optimal selling policies [4].

Models considering the point of view of car rental customers are still underexplored. A variant of the traveling salesman problem (*TSP*), named Traveling Car Renter Problem (*CaRS*) was proposed recently [8]. *CaRS* generalizes the *TSP* by allowing several cars with different costs to be available for the salesman. Therefore, different cars can be used during the salesman's tour. When a car is delivered to a city different from the one it was rented, an extra fee must be paid. The objective is to find the route and the sequence of rented cars that minimizes travel costs and extra fees. Since the *TSP* is NP-hard [5] and a special case of *CaRS* when the set of available cars has cardinality 1, *CaRS* is also NP-hard. Heuristic approaches proposed for *CaRS* include greedy randomized adaptive search, evolutionary algorithms and local search [3,8,9,21]. A model and evolutionary algorithms for the quota variant of *CaRS* were presented in [10]. Mathematical programming models, on the other hand, have not been fully explored.

In this study we present integer programming formulations for *CaRS*. Two formulations have quadratic objective functions and linear constraints. One formulations has a linear objective function and some quadratic constraints. The first formulation with a quadratic objective function is grounded on the *TSP* interpreted as a special case of the quadratic assignment problem (*QAP*) in which the assignment variables refer to visitation orders. The second formulation with a quadratic objective function is based on the Gavish and Grave's network flow model for the *TSP*. A model based on the Dantzig–Fulkerson–Johnson's formulation for the *TSP* is presented with quadratic constraints. Linearization for practical computational issues concerning the quadratic formulations are presented. The linearized models were implemented in two solvers. An experiment with 50 *CaRS* instances is reported. The integer formulations are presented in Sect. 2 and the experiments are presented in Sect. 3. Finally, conclusions are presented in Sect. 4.

2 Integer programs for *CaRS*

This section presents integer formulations for *CaRS*. Formulations with quadratic objective functions and linear constraints are presented in Sects. 2.1 and 2.2. A formulation with a linear objective function and quadratic constraints is presented in Sect. 2.3.

In this study, we consider a graph $G(N, A)$, where N is a set of n nodes (cities) and A is a set of arcs (roads). The salesman is considered to travel with rented cars. A set of cars, C , is available for the salesman's travel. Specific operational costs are associated with each car, including fuel consumption, toll payment and rent costs. When applied to arc $(i, j) \in A$, most of these specific costs are function of the distance to be traveled on that arc. There is an extra fee to be paid related to the company cost of taking a car from where it was delivered back to the city where it was rented. The problem is to find a minimal cost spanning cycle in G such that the travel costs are minimized, which includes the extra fees. The parameters of the formulations presented in this study are defined as follows.

Parameters

$|C|$: Cardinality of set C

d^c_{ij}: Total operational cost of driving car c, $c = 1, ..., |C|$, in arc $(i, j) \in A$

γ^c_{ij}: Fee to return car c from node j to node i, $i, j \in N, i \neq j$

2.1 Formulation based on the QAP

The formulation presented in this section is an extension of the *TSP* viewed as a *QAP*. Relationships among the *QAP*, *TSP* and *CaRS* are briefly commented further. A natural model for *CaRS* can be conceived as an integer quadratic programming problem, with a quadratic objective function and linear constraints. The variables of this formulation are defined as follows.

Variables

x^c_{ki}: Binary variable indicating whether car c arrives at node i in the k-th visitation order ($x^c_{ki} = 1$) or not ($x^c_{ki} = 0$)

y^c_i: Binary variable indicating whether car c is rented in node i ($y^c_i = 1$) or not ($y^c_i = 0$)

z^c_j: Binary variable indicating whether car c is delivered to node j ($z^c_j = 1$) or not ($z^c_j = 0$)

The quadratic car assignment and routing problem is described by objective function (1) and constraints (2)–(17). The model size depends essentially upon parameters n and $|C|$. It has $|C|\,n^2+2\,|C|\,n$ binary variables. A unique visitation order k, $k = 1, 2, ..., n$, is assigned to each node $i \in N$. The traveler is allowed to drive exactly one car $c \in C$ from city i to j, $i, j \in N$.

$$\min \sum_{c\in C}\left[\sum_{k=1}^{n-1}\sum_{(i,j)\in A}\sum_{c'\in C} d^c_{ij} x^c_{ki} x^{c'}_{(k+1)j} + \sum_{(1,j)\in A}\sum_{c'\in C} d^c_{1j} x^c_{n1} x^{c'}_{1j} + \sum_{i\in N, j\in N} \gamma^c_{ij} y^c_i z^c_j\right] \tag{1}$$

subject to

$$\sum_{c\in C}\sum_{k=1}^{n} x^c_{ki} = 1 \quad \forall\, i \in N \tag{2}$$

$$\sum_{c\in C}\sum_{i=2}^{n} x^c_{ki} = 1 \quad \forall\, k = 1, ..., n-1 \tag{3}$$

$$\sum_{c \in C} x_{n1}^{c} = 1 \tag{4}$$

$$\sum_{c \in C} y_{1}^{c} = 1 \tag{5}$$

$$\sum_{c \in C} z_{1}^{c} = 1 \tag{6}$$

$$\sum_{c \in C} y_{i}^{c} - \sum_{c \in C} z_{i}^{c} = 0 \quad \forall\, i \in N \backslash \{1\} \tag{7}$$

$$y_{i}^{c'} - \sum_{c \in C, c \neq c'} z_{i}^{c} \leq 0 \quad \forall\, i \in N \backslash \{1\},\ \forall\, c' \in C \tag{8}$$

$$z_{i}^{c'} - \sum_{c \in C, c \neq c'} y_{i}^{c} \leq 0 \quad \forall\, i \in N \backslash \{1\},\ \forall\, c' \in C \tag{9}$$

$$\sum_{j \in N} x_{1j}^{c} - y_{1}^{c} = 0 \quad \forall\, c \in C \tag{10}$$

$$\sum_{c \in C, c \neq c'} \sum_{k=1}^{n-1} x_{ki}^{c} - y_{i}^{c'} \geq 0 \quad \forall\, i \in N \backslash \{1\},\ \ \forall\, c' \in C \tag{11}$$

$$x_{ki}^{c'} - \sum_{j \in N} x_{(k+1)j}^{c'} - \sum_{c \in C, c \neq c'} y_{i}^{c} \leq 0 \ \ \forall i \in N \backslash \{1\}, \forall k = 1, ..., n-1, \forall c' \in C \tag{12}$$

$$x_{n1}^{c} - z_{1}^{c} = 0 \quad \forall\, c \in C \tag{13}$$

$$\sum_{k=1}^{n} x_{ki}^{c} - z_{i}^{c} \geq 0 \quad \forall\, i \in N \backslash \{1\},\ \ \forall\, c \in C \tag{14}$$

$$x_{ki}^{c} - \sum_{j \in N} x_{(k+1)j}^{c} - z_{i}^{c} \leq 0 \ \ \forall i \in N \backslash \{1\}, \forall k = 1, ..., n-1, \forall c \in C \tag{15}$$

$$x_{ki}^{c} \in \{0, 1\} \ \ \forall\, k = 1, ..., n,\ \forall\, i \in N,\ \forall c \in C \tag{16}$$

$$y_{i}^{c}, z_{j}^{c} \in \{0, 1\} \ \ \forall\, j \in N,\ \ \forall\, c \in C \tag{17}$$

The quadratic objective function (1) sums the operational costs incurred in the node visitation order plus extra return fees. The second parcel of expression (1) refers to the operational cost incurred in the first trip, that starts at the origin node, 1. For each car c, the third parcel of the objective function does not depend upon the path from node i to j, cities i and j being, respectively, the places where car c was rented and delivered.

Equations (2)–(4) are assignment constraints. Constraint (2) states that every city is visited in a unique order and with exactly one car. Constraint (3) states that a generic visitation order $k \neq n$ is assigned to each city different from the origin, 1. In complement, constraint (4) states that one car arrives at node 1 which is the last node to be visited. Equations (4)–(6) are specific to the origin node, 1, thus appearing only once in the model.

Expressions (5)–(9) refer to logical constraints concerning car rental and delivery variables y and z. Constraints (5) and (6) force a car rental and a car delivery to the origin node. Constraint (5) ensures that exactly one car is rented in city 1. Constraint (6) states that exactly one car is delivered to the origin city, 1. Constraints (7)–(9) couple variables y and z for the other nodes. If a car is rented in city $i, i \neq 1$, then some car is delivered there, as stated in (7). For any node $i \neq 1$, constraint (8) states that if car c' is rented in i then some other car $c, c \neq c'$, must be delivered to i. Constraint (9) states that, for any node $i \neq 1$, if a car c' is delivered to i then some other car $c, c \neq c'$, must be rented in i to go on the trip.

Constraints (10)–(15) couple variables x and y or x and z. Constraints (10) and (13) are specific for the starting node 1 and the last visitation order n, for each car $c \in C$. Constraint (10) ensures that if car c is rented in the starting node 1 then c is used in a first visitation order to some node j adjacent to 1. Constraint (13) ensures that the car that arrives at the starting city, closing the Hamiltonian cycle, is delivered to that city. Constraint (11) couples variables x and y and states that car c' can be rented in node i only if a different car $c, c \neq c'$, has arrived in some order at node i. Constraint (12) ensures that if car c' arrives in order k at node i then either c' is used to go on the trip to an adjacent node j or another car $c, c \neq c'$, is rented in node i. Constraint (14) couples variables x and z and ensures the delivery of car c to node i only if c has arrived at i in some order. Constraint (15) ensures that if car c arrives at node i in order k, then either c is used in a subsequent order to visit an adjacent node j or c is delivered to node i. At last, constraints (16) and (17) state that vectors x, y and z are binary.

The quadratic formulation presented for *CaRS* (1-17) is a direct extension of the *TSP* viewed as a *QAP* in the spirit of Koopmans and Beckmann [12]. The original *QAP* involves the assignment of industrial plants to locations, with an objective function that has a linear and a quadratic part. The interest of this study is related to the quadratic part of the *QAP*, referred as pure *QAP*. The input parameters of a pure *QAP* are restricted to two square matrices, B and D, of order n. The elements of matrix $B = [b_{kl}]$ are interpreted as flows between plants k and l, $k, l = 1, ..., n$, and the elements of matrix $D = [d_{ij}]$ are interpreted as distances between locations i and j, $i, j = 1, ..., n$. Variable p_{ki} is binary, $k, i = 1, ..., n$, and indicates whether plant k is located at i ($p_{ki} = 1$) or not ($p_{ki} = 0$). When plants k and l, $k \neq l$, are located in i and j, $i \neq j$, respectively, the unitary cost to transport the flow between k and l (b_{kl}) is directly proportional to distance d_{ij}. The objective function minimizes the total transportation cost of intermediate commodities, in such a way that the pure *QAP* formulation is expressed in (18)–(21).

$$\min \sum_{k=1}^{n} \sum_{i=1}^{n} b_{kl} p_{ki} d_{ij} p_{lj} \tag{18}$$

subject to

$$\sum_{k=1}^{n} p_{ki} = 1 \quad \forall\, i = 1, ..., n \tag{19}$$

$$\sum_{i=1}^{n} p_{ki} = 1 \quad \forall\, k = 1, \ldots, n \tag{20}$$

$$p_{ki} \in \{0, 1\} \quad \forall\, k, i = 1, \ldots, n \tag{21}$$

The formulation of the *TSP* as a *QAP* has the objective function (22) with constraints (19)–(21) and $p_{n1} = 1$.

$$\min \sum_{k=1}^{n-1} \sum_{(i,j)\in A} d_{ij}\; p_{ki}\; p_{(k+1)j} + \sum_{(1,j)\in A} d_{1j}\; p_{n1}\; p_{1j} \tag{22}$$

The *TSP* can be interpreted as a particular case of the *QAP* [12] when only n elements of matrix B are not null, with a set of 1s above the main diagonal and a coefficient 1 in the last position of the first column. In the words of Koopmans and Beckmann, the one and only intermediate commodity now is a traveling salesman who is required to call once at each location and return to his point of departure. Plant k of the original *QAP* is interpreted as the visitation order k in the *TSP*. The idea of flows among plants (orders) is maintained, but flows occur only among subsequent orders in the *TSP*. The starting point is node 1, that must be visited in the last order n, while each other node must be visited at some order k, $1 \le k < n$.

When the salesman has one car to make the tour, i. e., $C = \{c\}$, there is no fee to return c to its home city since $y_1^c = 1$ and $z_1^c = 1$. By making $x_{ki}^c = p_{ki}$, *CaRS* can be solved by the *TSP* formulation. The assignment equalities, (19) and (20), and $p_{n1} = 1$ ensure that constraints (2), (3) and (4) are satisfied. The other constraints, (5)–(17), are also trivially satisfied for a minimal cost tour with a single car. It is in this sense that the *CaRS* formulation (1-17) is a direct extension of the *TSP* interpreted as a *QAP* as presented by Koopmans and Beckmann [12].

2.1.1 A mixed integer linear program for CaRS

To obtain a linear formulation, the idea is to extend to *CaRS* a stronger variation of the linearization suggested by Koopmans and Beckmann for the *TSP*. Flow balance equations that enable a linear formulation of the *QAP* are presented in [12]. Equations for matrix B of the *TSP* are rewritten in terms of cars flow balancing. Exactly one car must arrive at each node and exactly one car must depart from each node. The car that comes into a node can be the same car that goes out from that node. Otherwise, the car that arrives at a node can be discarded and a new car must then be rented in this node to progress on the travel. The cars are driven by the salesman in such a way that each node is visited exactly once. In this context, the f_{hi}^{kc} and w_{ij}^c binary variables are defined as follows.

Variables

f_{hi}^{kc}: Binary variable indicating whether coming from node h, node i is visited in order k with car c ($f_{hi}^{kc} = 1$) or not ($f_{hi}^{kc} = 0$)

w_{ij}^c: Binary variable indicating whether car c is rented in node i and delivered to node j ($w_{ij}^c = 1$) or not ($w_{ij}^c = 0$)

$$\min \sum_{c\in C}\left[\sum_{k=1}^{n}\sum_{(h,i)\in A} d_{hi}^{c} f_{hi}^{kc} + \sum_{i\in N, j\in N} \gamma_{ij}^{c} w_{ij}^{c}\right] \tag{23}$$

$$\sum_{c\in C}\sum_{j\in N} f_{1j}^{1c} = \sum_{c\in C} x_{n1}^{c} \qquad \forall i \in N \tag{24}$$

$$\sum_{j\in N} f_{ij}^{kc} = x_{(k-1)i}^{c} \qquad \forall k = 2, ..., n,\ \forall i \in N,\ \forall c \in C \tag{25}$$

$$\sum_{h\in N} f_{hi}^{1c} = x_{1i}^{c} \qquad \forall i \in N,\ \forall c \in C \tag{26}$$

$$\sum_{h\in N} f_{hi}^{kc} = x_{ki}^{c} \qquad \forall k = 2, ..., n,\ \forall i \in N,\ \forall c \in C \tag{27}$$

$$\sum_{j\in N} w_{ij}^{c} = y_{i}^{c} \qquad \forall i \in N,\ \forall c \in C \tag{28}$$

$$\sum_{i\in N} w_{ij}^{c} = z_{j}^{c} \qquad \forall j \in N,\ \forall c \in C \tag{29}$$

$$w_{ij}^{c} \in \{0, 1\} \qquad \forall i, j \in N,\ \forall c \in C \tag{30}$$

$$f_{hi}^{kc} \in \{0, 1\} \qquad \forall i, j \in N,\ \forall k = 1, ..., n,\ \forall c \in C \tag{31}$$

The first two parcels of function (1) were replaced by the first parcel of function (23) where the f_{hi}^{kc} variable replaced the quadratic term $x_{(k-1)h}^{c'} x_{(k)i}^{c}$. The f_{hi}^{kc} variable can be interpreted as the flow from location h to location i of the commodity supplied by plant $k-1$ to plant k. The visitation order index of node h can be suppressed once h precedes i and if i is visited in the k-th order, then h is visited in the $(k-1)$-th order. In addition to the f_{hi}^{kc} flow variable, the original formulation (1)–(17) can be linearized with the introduction of $|C|\, n^2$ variables w_{ij}^{c} which replace y_i^c and z_j^c in the objective function. In accordance with the previous definitions of the rent and deliver variables, the equivalence relation $w_{ij}^{c} = y_i^c z_j^c$ is implicit in a linear model, in such a way that $w_{ij}^{c} = 1$ if and only if $y_i^c = 1$ and $z_j^c = 1$. In summary, the mixed integer linear program for the traveling car renter problem has the objective function (23) subject to constraints (2)–(17) and to additional constraints (24)–(31). Constraints (24)–(27) couple variables f_{hi}^{kc} and x_{ij}^{c}. Constraints (28) and (29) are due to the linearization.

2.2 Formulation based on network flow

The formulation presented in this section extends the model presented for the *TSP* in [6] and uses flow constraints. The problem is formulated in (32)–(43). The variables are defined as follows.

Variables

- f_{ij}^{c}: Binary variable indicating whether car c traverses edge (i, j) from i to j ($f_{ij}^{c} = 1$) or not ($f_{ij}^{c} = 0$)
- u_{ij}: Arbitrary non-negative integers

y_i^c: Binary variable indicating whether car c is rented in node i ($y_i^c = 1$) or not ($y_i^c = 0$)

z_j^c: Binary variable indicating whether car c is delivered to node j ($z_j^c = 1$) or not ($z_j^c = 0$)

Constraints (33)–(36) and (43) came from the formulation presented by Gavish and Graves [6] for the *TSP*. Constraints (33) and (34) ensure that only one car arrives in each city, coming from one arc, and only one car leaves each city using one arc. Constraints (35) and (36) form a network flow problem and were included to prevent subtours. The u_{ij} variable can be interpreted as the flow of a single commodity to vertex 1 from every other vertex [18]. The proof that these constraints prevent subtours is presented in [6]. Constraint (37) ensures that if car c left node j but did not arrive there, then c was rented in city j. The model does not prevent y_j^c from being set to 1 when car c goes through node j and is not rented there. However, this is never the case when it comes to the optimal solution. Constraint (38) ensures that a car is rented in 1. Similarly, constraints (39) and (40) ensure the car delivery. At last, constraint (42) ensures that variable f_{ij}^c is binary.

$$\min \sum_{c \in C} \left(\sum_{(i,j) \in M} d_{ij}^c f_{ij}^c + \sum_{c \in C} \gamma_{ij}^c y_i^c z_j^c \right) \tag{32}$$

subject to

$$\sum_{c \in C} \sum_{i \in N} f_{ij}^c = 1 \qquad \forall j \in N \tag{33}$$

$$\sum_{c \in C} \sum_{j \in N} f_{ij}^c = 1 \qquad \forall i \in N \tag{34}$$

$$\sum_{\substack{j \in N \\ i \neq j}} u_{ij} - \sum_{\substack{j \in N \setminus \{1\} \\ i \neq j}} u_{ji} = 1 \qquad \forall i \in N \setminus \{1\} \tag{35}$$

$$u_{ij} \leq (n-1) \sum_{c \in C} f_{ij}^c \qquad \forall (i,j) \in A, i \neq 1 \tag{36}$$

$$\sum_{\substack{i \in N, \\ i \neq j}} f_{ji}^c - \sum_{\substack{i \in N, \\ i \neq j}} f_{ij}^c \leq y_j^c \qquad \forall j \in N \setminus \{1\}, \quad \forall c \in C \tag{37}$$

$$\sum_{i \in N} f_{1i}^c = y_1^c \qquad \forall c \in C \tag{38}$$

$$\sum_{\substack{i \in N, \\ i \neq j}} f_{ij}^c - \sum_{\substack{i \in N, \\ i \neq j}} f_{ji}^c \leq z_j^c \qquad \forall j \in N \setminus \{1\}, \quad \forall c \in C \tag{39}$$

$$\sum_{i \in N} f_{i1}^c = z_1^c \qquad \forall c \in C \tag{40}$$

$$y_i^c, z_j^c \geq 0 \qquad \forall i, j \in N, \quad \forall c \in C \tag{41}$$

$$f_{ij}^c \in \{0, 1\} \qquad \forall (i, j) \in A, \quad \forall c \in C \tag{42}$$

$$u_{ij} \geq 0 \qquad \forall (i, j) \in A \tag{43}$$

A linearized formulation for *CaRS* can be derived from (32)–(43). The objective function is described in (44). The w_{ij}^c variable, as defined in Sect. 2.1.1, indicates whether car c is rented in node i and delivered to node j. The second parcel of function (32) is replaced by the second parcel of function (44). Constraint (45) is due to the linearization and constraint (46) ensures that the w_{ij}^c variable is binary.

$$\min \sum_{c \in C} \left(\sum_{(i,j) \in M} d_{ij}^c f_{ij}^c + \sum_{c \in C} \gamma_{ij}^c w_{ij}^c \right) \tag{44}$$

$$y_i^c + z_j^c - 1 \leq w_{ij}^c \quad \forall (i, j) \in A, \quad \forall c \in C \tag{45}$$

$$w_{ij}^c \in \{0, 1\} \quad \forall (i, j) \in A, \quad \forall c \in C \tag{46}$$

2.3 A formulation with quadratic constraints

The model presented in this section is based on the Dantzig–Fulkerson–Johnson's formulation for the TSP [2]. The mathematical formulation considers variables f_{ij}^c and u_i defined as follows. Variable w_{ij}^c is defined as in Sect. 2.1.1.

Variables

- f_{ij}^c: Binary variable indicating whether car c traverses edge (i, j) from i to j ($f_{ij}^c = 1$) or not ($f_{ij}^c = 0$)
- u_i: The order in which vertex i is visited on the tour.

$$\min \sum_{c \in C} \sum_{i,j \in N} d_{ij}^c f_{ij}^c + \sum_{c \in C} \sum_{i,j \in N} \gamma_{ij}^c w_{ij}^c \tag{47}$$

$$\sum_{c \in C} \sum_{j \in N} f_{ij}^c = 1 \qquad \forall i \in N \tag{48}$$

$$\sum_{c \in C} \sum_{i \in N} f_{ij}^c = 1 \qquad \forall j \in N \tag{49}$$

$$y_i^c = \left(\sum_{j \in N} f_{ij}^c \right) \left(\sum_{c' \in C, c' \neq c} \sum_{h \in N} f_{hi}^{c'} \right) \qquad \forall c \in C, \forall i \in N \tag{50}$$

$$z_i^c = \left(\sum_{j \in N} f_{ji}^c \right) \left(\sum_{c' \in C, c' \neq c} \sum_{h \in N} f_{ih}^{c'} \right) \qquad \forall c \in C, \forall i \in N \tag{51}$$

$$w_{ij}^c = y_j^c z_i^c \qquad \forall c \in C, \forall i, j \in N \tag{52}$$

$$\sum_{c \in C} y_1^c = 1 \tag{53}$$

$$2 \leq u_i \leq n \qquad \forall i \in N \setminus \{1\} \tag{54}$$

$$u_i - u_j + 1 \leq (n-1)(1 - \sum_{c \in C} f_{ij}^c) \qquad \forall i, j \in N \setminus \{1\} \tag{55}$$

$$f_{ij}^c, y_i^c, z_i^c \in \{0, 1\} \qquad \forall c \in C, \forall i, j \in N \tag{56}$$

$$u_i \in \aleph \qquad \forall i \in N \tag{57}$$

The linear objective function shown in (47) sums the costs of traversing edges with different cars and return fees. Equations (48) and (49) are assignment constraints and state that each vertex is visited once. Constraint (48) guarantees that only 1 car coming from 1 arc arrives at node i. Constraint (49) guarantees that only 1 car leaves node j passing by 1 arc. Constraint (50) states that if car c is rented in city i, an edge (i, j) must be traversed with car c and an edge (h, i) must be traversed with car, c', $c' \neq c$. Constraint (51) couples variables f_{ij}^c and z_i^c concerning car c delivered to node i. Constraint (52) sets variable w_{ij}^c according to the nodes where car c is rented and delivered. Constraint (53) ensures that one car is rented in node 1. Constraints (54) and (55) were adapted from the Miller-Tucker-Zemlin formulation for the *TSP* presented in [15]. These constraints prevent subtours. Constraint (54) ensures that vertex i is the u_i-th vertex visited on the tour. Since the problem requires that vertex 1 is visited first, it was removed from the set of nodes considered in constraint (54). Constraint (55) couples variables f_{ij}^c and u_i. Constraint (56) sets the binary variables and constraint (57) sets variables u_i to the range of positive integers.

In the models presented so far cars can be repeated along the tour. If car repetition is not allowed, it is necessary to add constraint (58) which ensures that each car can be rented at most once.

$$\sum_{i \in N} y_i^c \leq 1 \qquad \forall c \in C \tag{58}$$

2.3.1 A linearization for quadratic constraints

Constraints (50)–(52) presented in Sect. 2.3 are quadratic and their variables are binary. To work around this problem, we applied the usual linearization as described in [20] and reformulated in [7] where a non linear constraint, as in (59), is replaced by the set of equations (60)–(63).

$$s = q \times r \tag{59}$$

$$s \leq q \tag{60}$$

$$s \leq r \tag{61}$$

$$s \geq r + q - 1 \tag{62}$$

$$q, r, s \in \{0, 1\} \tag{63}$$

Constraints (50)–(52) are replaced by (64)–(72).

$$y_i^c \leq \left(\sum_{j \in N} f_{ij}^c \right) \quad \forall c \in C, \forall i \in N \quad (64)$$

$$y_i^c \leq \left(\sum_{c' \in C, c' \neq c} \sum_{h \in N} f_{hi}^{c'} \right) \quad \forall c \in C, \forall i \in N \quad (65)$$

$$y_i^c \geq \left(\sum_{j \in N} f_{ij}^c \right) + \left(\sum_{c' \in C, c' \neq c} \sum_{h \in N} f_{hi}^{c'} \right) - 1 \quad \forall c \in C, \forall i \in N \quad (66)$$

$$z_i^c \leq \left(\sum_{j \in N} f_{ji}^c \right) \quad \forall c \in C, \forall i \in N \quad (67)$$

$$z_i^c \leq \left(\sum_{c' \in C, c' \neq c} \sum_{h \in N} f_{ih}^{c'} \right) \quad \forall c \in C, \forall i \in N \quad (68)$$

$$z_i^c \geq \left(\sum_{j \in N} f_{ji}^c \right) + \left(\sum_{c' \in C, c' \neq c} \sum_{h \in N} f_{ih}^{c'} \right) - 1 \quad \forall c \in C, \forall i \in N \quad (69)$$

$$w_{ij}^c \leq y_j^c \quad \forall c \in C, \forall i, j \in N \quad (70)$$

$$w_{ij}^c \leq z_i^c \quad \forall c \in C, \forall i, j \in N \quad (71)$$

$$w_{ij}^c \geq y_j^c + z_i^c - 1 \quad \forall c \in C, \forall i \in N \quad (72)$$

$$w_{ij}^c \in \{0, 1\} \quad \forall c \in C, \forall i, j \in N \quad (73)$$

3 Experiments

The linear formulations presented in Sects. 2.1.1, 2.2 and 2.3.1 were implemented in two solvers: CPLEX (version 12.6.3.0) and Gurobi (version 6.5.2). The models were implemented with constraint (58), i.e., each car can be rented at most once.The tests were carried out on a PC with an Intel Core i7 3.45GHz x 8 and 32 Gb of RAM which ran Ubuntu 16.04 64 bits. The processing time was limited to 10,000 s. We present the results of computational experiments based on 50 CaRS instances, which are available at http://www.dimap.ufrn.br/lae/en/projects/CaRS.php. The instances are symmetric, i.e., $d_{ij}^c = d_{ji}^c$ for all $i, j \in V$ and $c \in C$ and the underlying graph is complete. The instances were divided into two classes: *E* and *NE*. Three groups of instances were created for each class: real, random, and tsplib-like. A primary edge weight matrix was created for each instance. The three groups differed in terms of the method used to generate the primary matrices. For the real instances, the edge weights of the primary matrix were taken from real maps. For the random instances, the weights were generated uniformly from the range of [10, 500]. For tsplib-like instances, the distance

Table 1 Results for E instances

Instance	n	$\|C\|$	Gurobi									CPLEX								
			KB			DFJ			GG			KB			DFJ			GG		
			Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap	Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap
BraRJ14e	14	2	294	3.74	0	294	0.41	0	294	2.87	0	294	2.71	0	294	1.04	0	294	3.66	0
Libia14e	14	2	730	1.83	0	730	0.62	0	730	0.50	0	730	2.31	0	730	0.72	0	730	0.60	0
Argent16e	16	2	955	76.30	0	955	0.81	0	955	3.50	0	955	12.87	0	955	0.94	0	955	7.77	0
EUA17e	17	2	912	32.48	0	912	0.85	0	912	3.76	0	912	15.26	0	912	0.97	0	912	2.12	0
Indones14e	14	3	799	6.75	0	799	0.47	0	799	0.36	0	799	4.42	0	799	0.87	0	799	0.76	0
Argelia15e	15	3	840	286.28	0	840	9.93	0	840	6.95	0	840	194.03	0	840	12.79	0	840	13.87	0
India16e	16	3	1035	88.92	0	1035	1.48	0	1035	1.00	0	1035	58.75	0	1035	2.20	0	1035	2.16	0
China17e	17	3	1003	662.49	0	1003	49.86	0	1003	436.81	0	1003	6838.43	0	1003	24.40	0	1003	388.73	0
Mexico14e	14	4	789	86.05	0	789	3.85	0	789	2.46	0	789	86.05	0	789	7.40	0	789	5.36	0
Sudao15e	15	4	823	154.35	0	823	1.55	0	823	6.79	0	823	170.86	0	823	6.20	0	823	18.16	0
Austral16e	16	4	1051	2474.94	0	1051	85.87	0	1051	44.14	0	1051	3165.30	0	1051	112.60	0	1051	107.44	0
Canada17e	17	4	1251	476.42	0	1251	39.10	0	1251	133.14	0	1251	2555.81	0	1251	33.82	0	1251	30.84	0
Arabia14e	14	5	851	32.23	0	851	1.93	0	851	7.53	0	851	22.14	0	851	2.95	0	851	2.35	0

Table 1 continued

Instance	n	$\|C\|$	Gurobi									CPLEX								
			KB			DFJ			GG			KB			DFJ			GG		
			Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap	Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap
Cazaq15e	15	5	904	368.66	0	904	20.65	0	904	17.98	0	904	854.09	0	904	16.28	0	904	14.40	0
Russia17e	17	5	1061	1703.70	0	1061	52.99	0	1061	284.16	0	1061	4172.13	0	1061	51.20	0	1061	167.58	0
BraAM26e	26	3	467	10000	8.13	467	9.44	0	467	228.02	0	498	10000	17.60	467	19.58	0	467	237.02	0
BraMG30e	30	3	634	10000	28.07	529	72.13	0	529	10000	2.07	558	10000	19.30	529	234.12	0	529	10000	3.77
att48eA	48	3	46320	10000	38.19	34571	10000	0.0029	34571	198.70	0	52310	10000	45.34	34571	1591.99	0	34729	10000	1.91
att48eB	48	4	80056	10000	64.33	34856	10000	1.79	34524	800.42	0	122137	10000	76.64	34744	10000	2.82	35101	10000	4.39
eil51eA	51	3	2783	10000	61.22	1365	10000	5.64	1363	10000	11.51	2963	10000	63.48	1365	10000	6.87	1468	10000	22.01
eil51eB	51	5	2480	10000	61.12	1479	10000	30.76	1455	10000	25.97	3070	10000	68.91	1703	10000	40.89	1826	10000	45.08
Berlin52eA	52	3	22817	10000	67.52	8948	7337.70	0	8948	4458.30	0	21839	10000	66.00	8948	10000	1.50	10084	10000	16.25
Berlin52eB	52	4	16705	10000	56.89	9109	10000	8.38	8739	4839.40	0	24495	10000	70.68	9899	10000	16.47	10277	10000	20.79
pr76eA	76	3	263933	10000	67.78	110871	10000	3.99	110095	10000	2.13	165221	10000	48.53	110542	10000	3.07	141415	10000	31.92
pr76eB	76	4	646540	10000	96.18	114697	10000	7.65	114649	10000	6.74	165537	10000	100	117114	10000	9.72	151009	10000	35.93

Table 2 Results for *NE* instances

Instance	n	$\|C\|$	Gurobi									CPLEX								
			KB			DFJ			GG			KB			DFJ			GG		
			Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap	Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap
BraRJ14n	14	2	167	2.74	0	167	0.53	0	167	46.58	0	167	7.73	0	167	0.57	0	167	13.30	0
Libia14n	14	2	760	3.24	0	760	0.53	0	760	947.74	0	760	9.84	0	760	0.39	0	760	212.28	0
Argent16n	16	2	894	4.39	0	894	0.46	0	894	2163.60	0	894	20.18	0	894	0.32	0	894	172.12	0
EUA17n	17	2	822	19.93	0	822	0.88	0	822	4519.79	0	822	15.38	0	822	1.34	0	822	1408.36	0
Indones14n	14	3	796	79.81	0	796	1.48	0	796	102.28	0	796	112.25	0	796	7.64	0	796	78.11	0
Argelia15n	15	3	863	1421.81	0	863	6.53	0	863	7420.33	0	863	8744.09	0	863	18.81	0	865	10000	8.48
India16n	16	3	985	351.20	0	985	1.69	0	985	60.25	0	985	269.30	0	985	6.96	0	985	43.20	0
China17n	17	3	918	4735.68	0	918	11.99	0	918	10000	8.82	918	764.84	0	918	34.82	0	918	10000	6.94
Mexico14n	14	4	902	1742.86	0	902	58.93	0	902	3318.62	0	902	8023.45	0	902	70.01	0	902	1940.90	0
Sudao15n	15	4	1020	2641.37	0	1020	19.48	0	1020	10000	14.31	1020	2102.60	0	1020	35.61	0	1020	2068.39	0
Australia16n	16	4	1061	7138.54	0	1061	26.6	0	1065	10000	14.55	1061	8019.03	0	1061	146.21	0	1061	10000	9.65

Table 2 continued

Instance	n	$\|C\|$	Gurobi									CPLEX								
			KB			DFJ			GG			KB			DFJ			GG		
			Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap	Sol	T(s)	Gap	Sol	T(S)	Gap	Sol	T(s)	Gap
Canada17n	17	4	1136	2152.19	0	1136	12.42	0	1136	5397.69	0	1136	10000	4.19	1136	40.68	0	1136	1013.04	0
Arabia14n	14	5	1026	1346.70	0	1026	29.38	0	1026	65.78	0	1026	1717.25	0	1026	61.86	0	1026	89.22	0
Cazaq15n	15	5	1043	4177.06	0	1043	106.36	0	1043	2331.15	0	1043	7930.04	0	1043	76.13	0	1043	4722.26	0
Russia17n	17	5	1126	10000	11.53	1094	116.77	0	1094	10000	17.36	1192	10000	22.38	1094	590.26	0	1094	10000	9.99
BraAM26n	26	3	202	3896.05	0	202	6.17	0	202	10000	1.48	204	10000	11.07	202	49.63	0	202	5155.95	0
BraMG30n	30	3	288	10000	18.05	271	652.54	0	276	10000	9.05	293	10000	21.29	271	2162.69	0	275	10000	11.24
att48nA	48	3	1892	10000	52.37	987	9695.00	0	996	10000	7.83	3453	10000	74.07	987	10000	1.17	991	10000	4.47
att48nB	48	4	3617	10000	82.52	799	10000	5.75	862	10000	23.43	3516	10000	82.29	808	10000	7.15	839	10000	21.71
eil51nA	51	3	2118	10000	58.87	1062	10000	2.73	1077	10000	15.50	1778	10000	51.04	1062	10000	3.83	1062	10000	14.48
eil51nB	51	5	4262	10000	82.63	911	10000	11.63	953	10000	20.14	4599	10000	84.07	949	10000	19.38	958	10000	20.99
Berlin52nA	52	3	4582	10000	75.18	1303	10000	1.30	1355	10000	14.09	5260	10000	78.39	1303	10000	1.67	1344	10000	13.48
Berlin52nB	52	4	3165	10000	78.19	879	10000	8.76	875	10000	18.85	3343	10000	79.70	881	10000	13.96	888	10000	20.42
pr76nA	76	3	2235	10000	94.80	1191	10000	1.17	1230	10000	9.84	9555	10000	100	1198	10000	1.96	1210	10000	8.67
pr76nB	76	4	4064	10000	70.01	1394	10000	5.23	1446	10000	14.32	6420	10000	100	1448	10000	14.78	2469	10000	50.01

Table 3 Summary of the computational results

	Gurobi			CPLEX		
	KB	*DFJ*	*GG*	*KB*	*DFJ*	*GG*
E						
#solved	15	18	**20**	15	**18**	16
min gap	8.13	**0.0029**	2.07	17.60	**1.50**	1.91
av gap	22	**8.32**	9.68	57.65	**11.62**	20.23
max gap	96.18	30.76	**25.97**	100	**40.89**	45.08
av. proc. time	4258.21	3107.59	**2459.07**	4726.21	**2884.80**	3640.11
#best	16	19	**25**	15	**25**	17
NE						
#solved	15	**18**	11	13	**17**	12
min gap	11.53	**1.17**	1.48	4.19	**1.17**	4.47
av gap	62.42	**5.22**	13.54	59.04	**7.99**	15.42
max gap	82.52	**11.63**	23.43	100	**19.38**	50.01
av. proc. time	5188.54	**3229.91**	6654.95	6309.44	**3332.16**	5876.69
#best	15	**24**	16	14	**25**	16

matrices of TSP instances [19] were used as primary matrices. For NE instances, the weights of edges corresponding to car c, $1 \leq c \leq |C|$, were generated uniformly from the range of $[1.4\omega_{ij}, 2.0\omega_{ij}]$, where ω_{ij} denotes the element in position (i, j) of the primary matrix. For E instances, a list of n integers, L_c, was given for each car. The weights of edges corresponding to car c, $1 \leq c \leq |C|$, were calculated using equation (74), where $d[i][j]$ is the entry in the i-th row and j-th column of the primary matrix.

$$d_{ij}^{c} = \frac{2L_c[i] + 3L_c[j]}{3} + d[i][j] \tag{74}$$

Tables 1 and 2 show the results for *E* and *NE* instances, respectively. Each line shows the name of the instance (*Instance*), the number of vertices (n), the number of cars ($|C|$), and the results obtained with each solver (*Gurobi* and *CPLEX*) for each model. Columns *KB*, *DFJ* and *GG* show, respectively, the results of the formulations presented in Sects. 2.1.1, 2.3.1 and 2.2. The *Sol* column shows the cost of the optimal or best integer solution found by the solver until it reached the processing time limit. The *gap* column shows the percentage deviation of the value shown in *Sol* from the lower bound implemented in the solver. The percentage deviations were calculated using equation (75), where *LB* denotes the value of the lower bound computed by the solver and *Sol* is the cost of the best integer solution. The *T(s)* column shows the processing time, in seconds, required by the solver.

$$gap = \frac{Sol - LB}{LB} \times 100 \tag{75}$$

The value “0” in column *gap* indicates that the instance was solved to optimality with the corresponding model and solver. The value “10000” in column *T(s)* indicates that the solver stopped due to the processing time limit.

Table 3 summarizes the computational results presented in Tables 1 and 2. The lines show the number of problems solved to optimality (*#solved*), the minimum (*min gap*), average (*av gap*) and maximum percentage deviation (*max gap*), the average processing time (*av. proc. time*) and the number of best solutions obtained with each model (*#best*). Instances were counted as “solved to optimality” when the solver finished its processing before reaching the processing time limit. Therefore, instances for which optimal solutions were found, but the solver did not finish their processing, e.g. Canada17n with *KB* model in CPLEX and China17n with *GG* model in Gurobi and CPLEX, were not counted as “solved to optimality”. The best results are shown in bold.

For the *E* instances, as shown in Table 3, the *GG* model implemented in the Gurobi solved more instances to optimality than the other models implemented in the same solver, reached the best values of the objective function, the best maximum percentage deviation and the best average processing time. The *DFJ* model implemented in the Gurobi reached the best minimum and average percentage deviation. For the CPLEX, the best results were produced by the *DFJ* model.

The *DFJ* model produced the best results in both solvers for the *NE* instances. 18 and 17 *NE* instances were solved to optimality with the *DFJ* model implementation in the Gurobi and CPLEX, respectively. The best average processing times, minimum, average and maximum percentage deviation were obtained with the *DFJ* model. More *NE* instances were solved to optimality when implemented with the *KB* model than with the *GG* model. In average, less processing time was required by the Gurobi to process the *NE* instances with the *KB* model than with the *GG* model.

The *NE* instances required, in general, more time to be processed than the *E* instances and were harder for the *KB* and the *GG* model implemented in the Gurobi. Although the *DFJ* model spent more time to process the *NE* instances than the *E* instances in both solvers, the average percent deviations for the *NE* instances were lower than for the *E* instances. We also observed an influence of the solver in the results for the *E* instances.

4 Conclusion

This study presented three quadratic formulations and linearizations proposals for *CaRS*. The first model was based on the Koopmans and Beckmann’s formulation (*KB*) for the *QAP*. The *KB* model has a quadratic objective function and linear constraints. The second model was based on the Gavish and Graves’ formulation (*GG*) for the *TSP* where flow constraints prevent subtours. The third model was based on the Dantzig–Fulkerson–Johnson’s (*DFJ*) formulation for the *TSP*. The *DFJ* model has a linear function and quadratic constraints. Linearizations were presented for the quadratic models. The linearized models were implemented in two solvers: Gurobi and CPLEX. An experiment with 50 *CaRS* instances, divided into 2 classes (*E* and *NE*), was reported.

The best results for the *E* instances were obtained with the linearized *GG* and *DFJ* models implemented in the Gurobi and the CPLEX, respectively. The linearized *DFJ* model implemented in both solvers produced the best results for the *NE* instances regarding number of problems solved to optimality, processing time and percentage deviation from the lower bound computed by the solvers.

Acknowledgements The researches of M. C. Goldbarg and E. F. G. Goldbarg are partially supported by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico), Brazil, under Grants 301845/2013-1 and 308062/2014-0. The research of L. Corrales is partially supported by CONICET (Consejo Nacional de Investigaciones Científicas Y Técnicas).

References

1. Czerny, A.I., Shi, Z., Zhang, A.: Can market power be controlled by regulation of core prices alone? An empirical analysis of airport demand and car rental price. Transp. Res. Part A **91**, 260–272 (2016)
2. Dantzig, G.B., Fulkerson, D.R., Johnson, S.M.: Solution of a large-scale traveling salesman problem. Oper. Res. **2**(4), 393–410 (1954)
3. Felipe, D., Goldbarg, E.F.G., Goldbarg, M.C.: Scientific algorithms for the car renter salesman problem. In: Proceedings of the IEEE CEC 2014 Congress on Evolutionary Computation, vol. 1, pp. 873–879 (2014)
4. Fontem, B.: An optimal stopping policy for car rental businesses with purchasing customers. Ann. Oper. Res. (2016). doi:10.1007/s10479-016-2240-2
5. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to eht Theory of NP-completeness. W. H. Freeman, San Francisco (1979)
6. Gavish, B., Graves, S.C.: The Travelling Salesman Problem and Related Problems, Working paper GR-078-78, Operations Research Center, Massachusetts Institute of Technology (1978)
7. Glover, F., Woolsey, E.: Converting the 0–1 polynomial programming problem to a 0–1 linear program. Oper. Res. **22**, 180–182 (1974)
8. Goldbarg, M.C., Asconavieta, P.H., Goldbarg, E.F.G.: Memetic algorithm for the traveling car renter problem: an experimental investigation. Memet. Comput. **4**(2), 89–108 (2012)
9. Goldbarg, M.C., Goldbarg, E.F.G., Asconavieta, P.H., Menezes, M.da S., Luna, H.P.L.: A transgenetic algorithm applied to the traveling car renter problem. Expert Syst. Appl. **40**, 6298–6310 (2013)
10. Goldbarg, M.C., Goldbarg, E.F.G., Menezes, M.S., Luna, H.P.L.: Quota traveling car renter problem: model and evolutionary algorithms. Inf. Sci. **367–368**, 232–245 (2016)
11. Guerriero, F., Olivito, F.: Revenue models and policies for the car rental industry. J. Math. Model. Algorithms **13**, 247–282 (2014)
12. Koopmans, T.C., Beckmann, M.: Assignment problems and the location of economic activities. Econometrica **25**, 53–76 (1957)
13. Lazov, I.: Profit management of car rental companies. Eur. J. Oper. Res. (2016). doi:10.1016/j.ejor.2016.08.064
14. Li, D., Pang, P.: Dynamic booking control for car rental revenue management: a decomposition approach. Eur. J. Oper. Res. **256**, 850–867 (2017)
15. Miller, C.E., Tucker, A.W., Zemlin, R.A.: Integer programming formulation of traveling salesman problems. J. ACM (JACM) **4**, 326–329 (1960)
16. Oliveira, B.B., Caravilla, M.A., Oliveira, J.F.: A relax-and-fix-based algorithm for the vehicle-reservation assignment problem in a car rental company. Eur. J. Oper. Res. **237**, 729–737 (2014)
17. Oliveira, B.B., Caravilla, M.A., Oliveira, J.F.: Fleet and revenue management in a car rental companies: a literature review and an integrated conceptual framework. Omega (2016). doi:10.1016/j.omega.2016.08.011
18. Öncan, T., Altinel, I.K., Laporte, G.: A comparative analysis of several asymetric traveling salesman problem formulations. Comput. Oper. Res. **36**, 637–654 (2009)
19. Reinelt, G.: TSPLIB—a traveling salesman problem library. ORSA J. Comput. **4**, 376–384 (1991)
20. Robert, F.: Applications de lálgebre de Boole en recherche opérationelle. Revue Française de Recherche Opérationelle **14**, 17–26 (1960)

21. Silva, A.R.V., Ochi, L.S.: An efficient hybrid algorithm for the traveling car renter problem. Expert Syst. Appl. **64**, 132–140 (2016)
22. The Global Car Rental Industry 2016–2021: Trends, Forecast and Opportunity Analysis 2016, February 2016, Lucintel