Received: 21 October 2022 | Revised: 12 April 2023 | Accepted: 21 June 2023 | Published on: 5 July 2023

DOI: 10.1002/net.22172

RESEARCH ARTICLE

WILEY

A new *MILP* formulation for the flying sidekick traveling salesman problem

Maurizio Boccia[1] | **Andrea Mancuso**[1] | **Adriano Masone**[1] | **Claudio Sterle**[1,2]

[1]Department of Electrical Engineering and Information Technology, University of Naples Federico II, Naples, Italy
[2]Istituto di Analisi dei Sistemi ed Informatica A. Ruberti, IASI-CNR, Rome, Italy

Correspondence
Adriano Masone, Department of Electrical Engineering and Information Technology, University of Naples Federico II, Naples, Italy.
Email: adriano.masone@unina.it

Funding information
Ministero dell'Università e della Ricerca (National Centre for Sustainable Mobility "CN MOST"), Grant/Award Number: MUR_CN00000023-CUP_UNINA_E63C22000930007

Abstract

Nowadays, truck-and-drone problems represent one of the most studied classes of vehicle routing problems. The Flying Sidekick Traveling Salesman Problem (*FS-TSP*) is the first optimization problem defined in this class. Since its definition, several variants have been proposed differing for the side constraints related to the operating conditions and for the structure of the hybrid truck-and-drone delivery system. However, regardless the specific problem under investigation, determining the optimal solution of most of these routing problems is a very challenging task, due to the vehicle synchronization issue. On this basis, this work provides a new arc-based integer linear programming formulation for the *FS-TSP*. The solution of such formulation required the development of a branch-and-cut solution approach based on new families of valid inequalities and variable fixing strategies. We tested the proposed approach on different sets of benchmark instances. The experimentation shows that the proposed method is competitive or outperforms the state-of-the-art approaches, providing either the optimal solution or improved bounds for several instances unsolved before.

KEYWORDS

branch-and-cut, drones, FS-TSP, MILP

1 | INTRODUCTION

A truck-and-drone system is a hybrid delivery system composed of trucks and drones. The two kinds of vehicles can serve customers in tandem or independently. However, the drone is not completely autonomous because of its limited payload and battery capacity. Thus, the truck acts also as a mobile depot for the drone, swapping its battery and providing the parcels to be delivered. The system is aimed at serving all the customers minimizing the overall delivery time. The usage of such delivery systems and the deriving routing problems represent cutting edge research topic for the Operations Research community and related literature has grown extremely fast, as witnessed by the survey works reported in [15, 23]. In particular, a bunch of variants of truck-and-drone routing problems has been proposed differing for the side constraints envisaging either the operating conditions or different hybrid truck-and-drone delivery system structures. For the first case, we mention, among the others, the variants involving multiple customers within each drone sortie [20], the dependence of the drone energy consumption on the carried weight [16], and the presence of no-fly zones [27]. For the second case, instead, we cite the variants involving multitruck and multidrone homogeneous and heterogeneous fleets [18, 26].

Generally speaking, a truck-and-drone system can be conceived as a two-echelon distribution system, whose main complexity issue is represented by the synchronization between the two kinds of involved vehicles. Such an issue characterizes all

the two-echelon distribution systems and it is declined in multiple ways depending on the specific features of the considered vehicles. On this basis, with reference to a truck-and-drone system, we can define two different operational schemes: a drone is launched from and recovered at the same location where the truck stops for the whole duration of the sortie; the truck launches a drone at a location and recovers it at a different location. It is easy to understand that the drone battery capacity, or in other words its endurance, differently affects these two operational schemes. Indeed, in the first case, the truck waits for the drone at a location and it is always available for the recovering operation. Thus, such a scheme requires no actual synchronization between the vehicles, but, on the other hand, it does not fully exploit the advantages of such a hybrid delivery system. Instead, the second scheme presents opposite features. Indeed, it has the advantage of simultaneously exploiting the truck and the drone for the delivery operations, but, on the other side, it requires a significant coordination between the two vehicles, which have to necessarily meet at the recovery location within a time period imposed by the drone endurance.

From the routing problem perspective, the first operational scheme configures a truck-and-drone problem which partially recalls the ideas underlying the *p*-median path problem [1] and the truck-and-trailer routing problem [7]. On the other side, the second operational scheme matches the definition of the first truck-and-drone routing problem defined in the literature [17], referred to as the Flying Sidekick Traveling Salesman Problem (*FS-TSP*).

In short, the *FS-TSP* operations and main assumptions can be summarized as follows. A set of customers must be served by a hybrid delivery system combining a truck and a drone, both placed at a starting depot. The truck plays a twofold role: it serves the customers met along its route and acts as a mobile depot for the drone. More in detail, the truck carries the drone to a location where it is loaded with a package, and it takes off to serve a customer. While the drone is performing the delivery, the truck either serves other customers or directly moves to the landing location where the drone will be picked-up after completing the delivery. Without loss of generality, the take-off and landing locations correspond to the customer locations. Drone delivery operations are subject to two main restrictions: limited flight range, due to its battery endurance, and limited load capacity, since it can transport only one package at a time. For safety reasons, the drone can land only at the depot or on the top of the truck. Therefore, the drone must hover at a landing location until the truck arrives, so requiring the synchronization of their movements. After landing on the truck, the drone battery is replaced with a fully charged one, thus it can start a new delivery without idle time. The set of all the operations performed by the drone (take-off, delivery, and landing), is referred to as sortie in the literature. The objective of the system is to minimize the time required for the completion of the delivery operations.

For the sake of comprehension, Figure 1 sketches three different solutions of the *FS-TSP* on a small instance with one depot and five customers, represented by square and circles, respectively. Take-off and landing nodes are highlighted in green and red, respectively. The solid arrow indicates the movement of the truck (both with and without the drone on board), while the dashed arrow indicates the movement of the drone in flight. The values on the arcs represent the travel times. For the sake of simplicity, it is assumed that the two vehicles move at the same speed. The time required to complete the delivery operations corresponds to the time when both vehicles return to the depot, after having served all the customers. It can be calculated as the sum of two components: the routing time of the truck moving with the drone on board and the time spent by the drone to complete each sortie. For example, in the first solution, the truck moves with the drone on board to serve customer 1 with a travel time of 12. At node 1, the drone is launched to serve customer 2 while the truck serves customers 3 and 4. Then, the truck serves customer 5 where it also picks-up the drone. The duration of the drone path is 21 while that of the truck is 17. Thus, because of synchronization constraints, the duration of the drone sortie is equal to the maximum between these two values. Finally, the two vehicles return together to the depot with a travel time of 9. Consequently, the completion time will be $12 + 21 + 9 = 42$. Coherently with these considerations, it is possible to compute the travel time of the two vehicles, the duration of the sorties, and the delivery time for the other two solutions. It is also important to underline that the validity of such solutions depends on the endurance of the drone. If the drone endurance is lower than the duration of a sortie, such a sortie will be unfeasible, thus making the corresponding solution unfeasible as well. For example, if the drone endurance was equal to 20, the only feasible solution would be the third one.

It is straight to understand that the *FS-TSP* represents a more complex variant of the *TSP* which includes synchronization requirements. Thus, it is a NP-hard problem as well, and it is extremely difficult to optimally and/or effectively solve even on instances with few customers. As a result of this complexity, most contributions in literature tackle the *FS-TSP* by heuristic approaches [3, 9, 12], while just a few mathematical programming exact methods have been developed. Moreover, as will be highlighted in the next section, available exact methods either suffer of dimensionality drawbacks due to the usage of an exponential number of variables, or suffer of tightness deficiency due to the weakness of formulation constraints. In particular, formulations characterized by path-based variables generally belong to the first class of methods while formulations using three-index and/or time variables in combination with *big-M* constraints usually fall in the second one. This work is aimed at overcoming these inconveniences by providing the following threefold methodological contribution:

- We introduce a new mixed-integer linear programming (*MILP*) arc-based formulation for the *FS-TSP* with a polynomial number of three-index variables.

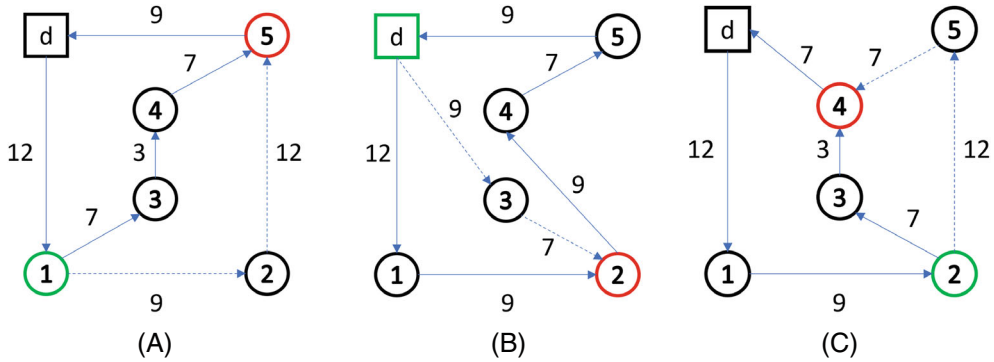

FIGURE 1 Three different flying sidekick traveling salesman problem solutions. (A) Delivery time = 42; (B) Delivery time = 46; (C) Delivery time = 47.

- We present a Branch-and-Cut (*B&C*) algorithm for solving the *FS-TSP* based on new families of valid inequalities and different variable fixing strategies. In particular, the valid inequalities are grounded on the vehicle coordination while the variable fixings are based on drone endurance.
- We provide an extensive experimentation on the benchmark instances used by the state-of-the-art algorithms for the *FS-TSP* [4, 11, 22]. The computational results show that our approach is competitive or outperforms the state-of-the-art approaches. Indeed, it either provides the optimal solution for some instances that have never been solved before or returns improved upper bounds to those that are still unsolved.

The remainder of the paper is organized as follows: in Section 2, we revise exact solution methods for the *FS-TSP*; in Section 3, we recall the *FS-TSP* and we present the *MILP* formulation; Section 4 describes the Branch-and-Cut algorithm, the valid inequalities, and the variable fixings; Section 5 is devoted to the computational results; finally, conclusions are given, and perspectives on future works on this topic are discussed in Section 6.

2 | LITERATURE REVIEW

The research activity on truck-and-drone delivery systems and, in particular, related routing problems have been significant in the last few years. For a complete review on variants of the *FS-TSP* the interested reader is referred to [16, 19], on single-truck multidrone to [6, 18], and on multitruck single/multidrone to [24, 26]. In this work, we focus our revision on exact solution methods for the *FS-TSP*. We describe each method from a twofold perspective to evaluate its effectiveness. From a mathematical point of view, the evaluation is based on the kinds of variables and constraints used to formulate the problem. In particular, arc-based formulations generally present a polynomial number of two/three-index variables, while path-based formulations use an exponential number of variables. The first kind of formulations is generally easy to implement within currently available solver frameworks, while the second one involves the development of ad hoc solution methods. Moreover, some formulations address the time dimension and/or the synchronization between the two vehicles by using *big-M* constraints that generally lead to weak linear relaxations. From a computational point of view, we highlight the instances used for the experimentation and briefly describe the resulting performance since the instance size and the computational burden are aspects generally used to evaluate a solution method. On this basis, we point out that the discussed methods consider one or more of the following three test beds for the *FS-TSP*:

- The set of 72 instances proposed in [17] with 10 customers and different endurance values (20 and 40 minutes) (*M10-instances*).
- The set of 240 instances proposed in [17] for a different truck-and-drone problem and adapted in [11] for the *FS-TSP*. These instances have 20 customers, and the endurance is set to either 20 or 40 min (*M20-instances*).
- The set of 300 instances proposed in [21] for a variant of the *FS-TSP* with up to 39 customers and an endurance equal to 20 min (*POI-instances*).

The first work addressing the *FS-TSP* is reported in [17]. In this work, the authors present for the first time the *FS-TSP* and propose a *MILP* model to solve it which uses three-index and time variables and *big-M* constraints. None of the *M10-instances* is solved by a commercial *MILP* solver within a time limit of 1800 seconds.

Starting from the *MILP* proposed in [17], a two- and a three-index formulation with time variables and *big-M* constraints are presented in [10] and tested on the *M10-instances*. The best results are obtained using the two-index formulation which solves 33 and 26 instances with a drone endurance of 20 and 40 min, respectively. The same authors successively propose an

improved formulation with only two-index variables and *big-M* constraints in [11], able to optimally solve for the first time all *M10-instances*. In addition, they test the new formulation on the *M20-instances*, providing the optimal solution of 58 and 2 instances with a drone endurance equal to 20 and 40 min, respectively. Recently, the same authors present a solution method based on a branch-and-bound scheme coupled with an *assignment problem* in [9]. They are able to solve only the *M10-instances* and a few instances with up to 19 customers of the *POI-instances*.

Almost contemporarily to [11], two new formulations for the *FS-TSP* are proposed in [25]. Two-index and time variables characterize both formulations, but the second one uses a joint representation of the drone sortie to reduce the number of variables. They solve to optimality: all the *M10-instances*; 32 *M20-instances* with an endurance equal to 20 min; 3 *M20-instances* with an endurance equal to 40 min.

A new formulation based on an extended graph representation of the *FS-TSP* with path-based variables but without *big-M* constraints is proposed in [4]. The authors develop a column-and-row generation approach to solve the proposed formulation. They are able to solve all the *M10-instances*, and 80 and 5 *M20-instances* of the considering a drone endurance equal to 20 and 40 min, respectively. The same authors then extend their method in [5] and [3], including cyclic sortie and trying to use the proposed method in combination with data science techniques, respectively.

The method currently able to optimally solve the largest *FS-TSP* instances is presented in [22]. The authors propose a compact formulation considering the truck and drone as a single entity leading to an exponential number of variables. Therefore, the authors develop a branch-and-price method to solve the formulation where the pricing problem is solved using a dynamic programming approach. The method is tested on a modified version of the *POI-instances*. In particular, the authors round up all the travel time to make them integer. The results show that the proposed method solved almost all the instances with up to 39 customers.

From this review, it is clear that our method is the only one considering an arc-based formulation with a polynomial number of three-index variables and without time variables and *big-M* constraints.

3 | *FS-TSP* DESCRIPTION AND FORMULATION

In this section, we first briefly recall the *FS-TSP* description as defined in [4, 11, 17, 22] and simultaneously introduce the problem notation, then we present the proposed original formulation.

3.1 | *FS-TSP* description and notation

The *FS-TSP* considers a set of customers, C, that have to be served exactly once, either by the truck or the drone. The two vehicles depart from and return to a single depot once. The proposed formulation splits the depot into a source node, s, and a destination node, t. Moreover, let V be the set of nodes, $V = C \cup \{s,t\}$, and A the set of arcs, $A = \{(i,j) : i \in C \cup \{s\}, j \in C \cup \{t\}\}$. The travel time of the truck (drone) on each arc (i,j), $(i,j) \in A$, is indicated with $t_{ij}(d_{ij})$.

The truck has an infinite capacity and serves all the customers met along its route. Moreover, it acts as a mobile depot for the drone providing the packages and dispatching the drone to serve the customers.

The drone can serve one customer per sortie. Therefore, a sortie is characterized by a launch node, a rendezvous node, and a customer node. The launch and the rendezvous nodes can be either the depot or a customer node. In addition, they must be different except if there is only one sortie starting from and ending at the depot. The duration of a sortie is limited by the drone endurance (*Dtl*). The truck must collect the drone before its flight time exceeds its endurance because the drone cannot land due to safety reasons. On this basis, it is clear that the two vehicles must be synchronized at the beginning and at the end of a sortie. Moreover, setup times arise when the drone is launched (launch time, *SL*) or retrieved (recovery time, *SR*). We point out that there is no launch time in the depot and that the recovery time is included in the endurance computation. For the sake of readability, the notation is also reported in Table 1.

The objective is to minimize the duration of the delivery process, that is, the time needed by the two vehicles to serve all the customers and return to the depot.

On the basis of this description, the *FS-TSP* involves three kinds of decisions: identification of the customers served by the drone; selection of launch and rendezvous nodes for each drone sortie; definition of the truck route.

3.2 | Problem formulation

To model the *FS-TSP* by a *MILP* formulation, we introduce the following sets of variables based on the previous problem description and notation:

- y_{ij}, $(i,j) \in A$, is equal to 1 if the truck travels along the arc (i,j) (with or without the drone on board), 0 otherwise.
- x_{ij}, $(i,j) \in A$, is equal to 1 if the drone flies along the arc (i,j), 0 otherwise.

TABLE 1 Notation used for the flying sidekick traveling salesman problem formulation.

Problem notation	
Sets:	
C	Set of customers
$\{s\}$	Tandem starting node
$\{t\}$	Tandem ending node
V	Set of nodes, $C \cup \{s,t\}$
A	Set of arcs
Parameters:	
t_{ij}	Truck travel time on the arc (i,j)
d_{ij}	Drone travel time on the arc (i,j)
Dtl	Drone endurance
SL	Launch service time
SR	Recovery service time

- γ_{ij}^h, $(i,j) \in A$, $h \in C$, is equal to 1 if the arc (i,j) is traveled by the truck while the drone is performing the drone sortie serving customer h, 0 otherwise.
- θ^h, $h \in C$, is equal to 1 if customer h is served by the drone, 0 otherwise.
- ω_i^h, $i \in C \cup \{s\}, h \in C$, is equal to 1 if node i is the launch node of the drone sortie serving customer h, 0 otherwise.
- δ_j^h, $j \in C \cup \{t\}, h \in C$, is equal to 1 if node j is the rendezvous node of the drone sortie serving customer h, 0 otherwise.
- σ^h, $h \in C \cup \{t\}$, is a continuous variable indicating the truck waiting time at the destination node of the sortie serving customer h. If customer h is served by the drone, then $\sigma^h = 0$.

Consistently with this notation and variables, the makespan of the delivery process, that is, the objective function, can be written as:

$$\min \sum_{(i,j)\in A} t_{ij} y_{ij} + \sum_{h\in C} (\mathrm{SL} + \mathrm{SR})\, \theta^h - \sum_{h\in C} \mathrm{SL}\, \omega_s^h + \sum_{h\in C\cup\{t\}} \sigma^h,$$

where the first term measures the duration of the truck path, the second and the third term evaluate the total launch and recovery service time (the launch service time at the depot is equal to 0), and the fourth term assesses the total truck waiting time.

Then, the set of constraints can be divided into the following six families.

Truck routing constraints

$$\sum_{j:(s,j)\in A} y_{sj} = \sum_{i:(i,t)\in A} y_{it} = 1. \tag{1}$$

$$\sum_{j:(i,j)\in A} y_{ij} = \sum_{j:(j,i)\in A} y_{ji} \le 1 \qquad i \in C. \tag{2}$$

$$\sum_{i,j\in S|(i,j)\in A} y_{ij} \le \sum_{h\in S\setminus\{q\}} (1-\theta^h) \qquad 2 \subseteq S \subseteq V,\ \ q \in S \setminus \{s,t\}. \tag{3}$$

They impose that there is a truck path from the origin node s to the destination node t, in any feasible solution. Constraints (3) are the subtour elimination constraints. As in the formulation of the *orienteering problem* [13], given a subset of nodes S, constraints (3) impose that the number of arcs with origin ad destination in S traveled by the truck must be less than the number of nodes served by the truck in the subset S.

Truck path and launch/rendezvous linking constraints

$$\sum_{j:(s,j)\in A} \gamma_{sj}^h = \omega_s^h \qquad h \in C. \tag{4}$$

$$\sum_{j:(i,t)\in A} \gamma_{it}^h = \delta_t^h \qquad h \in C. \tag{5}$$

$$\sum_{j:(i,j)\in A} \gamma_{ij}^h - \sum_{j:(i,j)\in A} \gamma_{ji}^h = \omega_i^h - \delta_i^h \qquad i,h \in C. \tag{6}$$

They impose that there is a truck path without the drone onboard from the launch node to the rendezvous node of each drone sortie serving customer h. The constraints (4) ensure that if a sortie includes the tandem starting node, then the launch node of the sortie must be the starting node itself. Similarly, the constraints (5) ensure that a sortie must finish at the tandem

ending node if it is part of the sortie. Lastly, the constraints (6) guarantee that a sortie must start at a launch node and end at a rendezvous node.

Single assignment constraints

$$y_{sj} + x_{sj} \leq 1 \qquad (s,j) \in A. \tag{7}$$

$$y_{it} + x_{it} \leq 1 \qquad (i,t) \in A. \tag{8}$$

$$y_{ij} + x_{ij} + x_{ji} \leq 1 \qquad i,j \in C | (i,j) \in A. \tag{9}$$

$$\sum_{j:(h,j)\in A} y_{hj} + \theta^h = 1 \qquad h \in C. \tag{10}$$

Constraints (7) and (8) impose that each arc cannot be traveled by the truck and the drone simultaneously for the arcs exiting from the starting node and for the arcs entering into the ending node, respectively. Constraints (9) impose that each arc between two customers can not be traveled by the truck and the drone simultaneously and avoid loops in which the launch and recovery nodes of a sortie are the same node. Constraints (10) impose that each customer must be served either by the truck or by the drone.

Consistency constraints

$$\sum_{h\in C} \gamma_{ij}^h \leq y_{ij} \qquad (i,j) \in A. \tag{11}$$

$$\sum_{i\in V\setminus\{t,h\}} \omega_i^h = \sum_{j\in V\setminus\{s,h\}} \delta_j^h = \theta^h \qquad h \in C. \tag{12}$$

$$x_{ij} \leq \theta^i + \theta^j \qquad (i,j) \in A. \tag{13}$$

$$x_{ij} \leq \omega_i^j + \delta_j^i \qquad (i,j) \in A. \tag{14}$$

$$\sum_{j:(i,j)\in A} x_{ij} = \sum_{h\in C\setminus\{i\}} \omega_i^h + \theta^i \leq 1 \qquad i \in C. \tag{15}$$

$$\sum_{i:(i,j)\in A} x_{ij} = \sum_{h\in C\setminus\{j\}} \delta_j^h + \theta^j \leq 1 \qquad j \in C. \tag{16}$$

Constraints (11) ensure the consistency between the arcs traveled by the truck during a sortie and the origin-destination truck path. Moreover, if a customer h is served by the drone ($\theta^h = 1$), constraints (12) impose that the corresponding sortie must have a launch and a rendezvous node. Constraints (13) guarantee that if the drone travels the arc $(i,j), (i,j) \in A$, then either node i or node j is a customer served by the drone. Constraints (14) ensure that if the drone travels the arc $(i,j), (i,j) \in A$, then either node i is the launch node of the sortie serving node j or node j is the rendezvous node of the sortie serving node i. Constraints (15) guarantee that an arc coming out of a node i can be traveled by the drone only if the drone serves i or it is the launch node of a sortie. Finally, constraints (16) ensure that an arc entering a node j can be traveled by the drone only if the drone serves j or it is the rendezvous node of a sortie.

Drone endurance constraints

$$\sum_{(i,j)\in A} t_{ij}\gamma_{ij}^h \leq (Dtl - SR)\theta^h \qquad h \in C. \tag{17}$$

$$\sum_{i\in V\setminus\{t\}} d_{ih}\omega_i^h + \sum_{j\in V\setminus\{s\}} d_{hj}\delta_j^h \leq (Dtl - SR)\theta^h \qquad h \in C. \tag{18}$$

Constraints (17) impose an upper bound on the duration of the truck path of the sortie serving customer h, while constraints (18) impose the same upper bound on the duration of the drone path of the sortie.

Waiting time constraints

$$\sum_{i\in V\setminus\{t\}} d_{ih}\omega_i^h + \sum_{j\in V\setminus\{s\}} d_{hj}\delta_j^h - \sum_{(i,j)\in A} t_{ij}\gamma_{ij}^h \leq \sigma^h \qquad h \in C. \tag{19}$$

Constraints (19) are needed to consider the truck waiting time in the objective function.

Based on the model just described, it is possible to highlight the main differences with the formulations previously described in the literature review. Specifically, the proposed model differs in three main aspects: the expression of the objective function, the subtour elimination constraints, and the constraints that ensure synchronization between the two vehicles.

Formulations that use time-variables [10, 22, 25] have such variables in each of these elements of the formulation. Specifically, the subtour elimination constraints and synchronization constraints are Big-M constraints. Therefore, these formulations are generally characterized by a weak lower bound at the root node. In contrast to the previous ones, the formulations proposed

in [11, 25] use time-variables, but these are only used for synchronization while the objective function is expressed as the sum of travel times and waiting times at the rendezvous nodes. This still involves the presence of Big-M constraints, but in a smaller number and therefore a better value of the lower bound.

Finally, the main difference with the formulation proposed in [4] is represented by the number of variables and constraints. Specifically, this formulation involved the use of an exponential number of variables for the sorties, as opposed to the polynomial order γ_{ij}^h variables of the proposed formulation. It is clear that the presence of an exponential number of variables also entails the presence of an exponential number of constraints to ensure consistency with the rest of the model. Such constraints are not necessary in the proposed model, and therefore the only exponential number of constraints are the subtour elimination constraints present also in the formulation reported in [4].

4 | BRANCH-AND-CUT APPROACH, VALID INEQUALITIES AND VARIABLE FIXINGS

The proposed formulation contains an exponential number of constraints due to the subtour elimination constraints (3). Therefore, we developed a Branch-and-Cut (*B&C*) procedure that solves the problem without constraints (3) and then adds to the formulation only the *lazy constraints* (3) violated each time an integer solution is found. Moreover, the *B&C* effectiveness is further improved by the integration of several valid inequalities and variable fixings. The valid inequalities are introduced to strengthen the proposed formulation, increasing the quality of the lower bound. The variable fixings are implemented to reduce the solution space. In this section, we begin by introducing the valid inequalities, then discuss the variable fixing strategies. Finally, we present the implementation settings for the *B&C*.

4.1 | Valid inequalities

This subsection presents four families of valid inequalities based on the coordination of the two vehicles (i.e., the coherency between the truck and the drone movements when operating in tandem).

4.1.1 | Cut inequalities

Proposition 1. *Given the graph $G(V,A)$ and two nodes p and q, $p,\ q \in V$, let $Cut(p,q)$ be a generic cut on the graph G between the nodes p and q, and let $h, h \in C$, be a generic customer. Then, the cut inequalities:*

$$\sum_{(i,j)\in Cut(s,h)} y_{ij} \geq 1 - \theta^h, \tag{20}$$

and

$$\sum_{(i,j)\in Cut(h,t)} y_{ij} \geq 1 - \theta^h, \tag{21}$$

are valid for the FS-TSP formulation.

Proof. If the drone does not serve customer h ($\theta^h = 0$), there must be a truck path from the origin node s to h and from h to the destination node t in any feasible solution. Therefore, an arc must cross any cut from s to h and from h to t. ∎

It is also simple to prove that constraints (20) and (21) can be used as *subtour elimination* constraints. Indeed, they guarantee that the set of the arcs traveled by the truck, $\{(i,j) \in A \mid y_{ij} = 1\}$, define a connected graph and constraints (2) prevent the truck from coming back to a previously visited node. So, any solution presenting a subtour violates at least an inequality in (20), (21), or (2). The cut inequalities (20) and (21) are separated and added to the formulation at each node belonging to the first three levels of the enumeration tree. The separation procedure solves the *minimum cut problem* [14], from the origin node s to h and from h to the destination node t on the weighted graph $G'(V,A,W)$ where the weight w_{ij} on the generic arc $(i,j), (i,j) \in A$, is given by the value of the corresponding y_{ij} variable. If the value of the minimum cut is less than $1 - \theta^h$, then the violated cut inequality is added to the formulation.

Proposition 2. *Let h and k be two customer nodes, $h,\ k \in C$. Let $Cut(h,k)$ and $Cut(k,h)$ be a cut between the nodes h and k and nodes k and h, respectively. The 2-cut inequality:*

$$\sum_{(i,j)\in Cut(h,k)} y_{ij} + \sum_{(i,j)\in Cut(k,h)} y_{ij} \geq 1 - \theta^h - \theta^k, \tag{22}$$

is valid for the FS-TSP formulation.

Proof. If customers h and k are not served by the drone ($\theta^h = \theta^k = 0$), then the truck either travels from h to k or it travels from from k to h. Therefore, a cut from h to k or from k to h must be crossed by an arc traveled by the truck. ■

As the previous set of cut inequalities, the 2-cut inequalities (22) are separated and added to the formulation at each node belonging to the first three levels of the enumeration tree. The separation procedure solves the *minimum cut problem* from the k to h and from h to k on the weighted graph $G'(V, A, W)$ where the weight w_{ij} on the generic arc $(i,j), (i,j) \in A$, is given by the value of the corresponding y_{ij} variable. If the value of the minimum cut is less than $1 - \theta^h - \theta^k$, then the violated cut inequality is added to the formulation.

Finally, the following proposition defines another family of cut valid inequalities by considering the sets of ω and δ variables.

Proposition 3. *Given a couple of nodes p and q, $p, q \in V$, and a customer node h, $h \in C$, the following inequality:*

$$\sum_{(i,j)\in Cut(p,q)} y_{ij} \geq \omega_p^h + \delta_q^h - 1, \tag{23}$$

is valid for the FS-TSP formulation.

Proof. If customer h is served by a drone launched from node p and retrieved at node q ($\omega_p^h = \delta_q^h = 1$), then the truck travels from p to q. Therefore, a cut from p to q must be crossed by an arc traveled by a truck. ■

The cut inequalities (23) are separated and added to the formulation at each node belonging to the first three levels of the enumeration tree. The separation procedure solves the *minimum cut problem* from the origin node p to the destination node q on the weighted graph $G'(V, A, W)$ where the weight w_{ij} on the generic arc $(i,j), (i,j) \in A$, is given by the value of the corresponding y_{ij} variable. If the value of the minimum cut is less than $\omega_p^h + \delta_q^h - 1$, then the violated cut inequality is added to the formulation.

4.1.2 | Drone sortie inequalities

Let *i-k-j* be a triple of nodes in the set V, if $d_{ik} + d_{kj} \geq Dtl - Sr$, then the sortie is infeasible and the inequality:

$$\omega_i^k + \delta_j^k \leq 1, \tag{24}$$

is valid for the *FS-TSP* formulation. Valid inequalities (24) are polynomial in number since the number of triple of nodes is $|V|^3$. As a result, these inequalities can be included at the beginning of the formulation.

4.1.3 | Truck sortie inequalities

Let A' be a subset of arcs, $A' = \{(i,j), (i,j) \in A : t_{ij} \geq Dtl - SR\}$, if $\omega_i^k = 1$, then $\sum_{l:(l,j)\in A'} \gamma_{lj}^k = 0$ for each $k \in C$ and for each $l \in V$. Therefore, the inequality:

$$\sum_{l:(l,j)\in A} \gamma_{lj}^k \leq 1 - \omega_i^k, \tag{25}$$

is valid for the *FS-TSP* formulation.

Likewise, if $\delta_j^k = 1$ then $\sum_{l:(i,l)\in A'} \gamma_{il}^k = 0$ for each $k \in C$ and for each $i \in V$. Therefore, the inequality:

$$\sum_{l:(i,l)\in A} \gamma_{il}^k \leq 1 - \delta_j^k, \tag{26}$$

is valid for the *FS-TSP* formulation.

Valid inequalities (25,26) are polynomial in number since the number of arcs is approximately $|V|^2$. As a result, these inequalities can be included as user cuts at the beginning of the formulation.

Let $P(i,j)$ be a path from i to j, $(P(i,j) = \{(i,s_1), (s_1,s_2), ..., (s_{n-1},j)\}$, and $D(P(i,j))$ be its track travel time, $D(P(i,j)) = \sum_{(i,j)\in P(i,j)} t_{ij}$. If $D(P(i,j)) > Dtl - SR$ then the inequalities:

$$\sum_{(i,j)\in P(i,j)} \gamma_{ij}^k \leq |P(i,j)| - 1 \quad k \in C, \tag{27}$$

are valid for the *FS-TSP* formulation.

Valid inequalities (27) are exponential in number since the number of possible paths is exponential to the number of nodes. Therefore, we add to the formulation only those for which $|P(i,j)| \leq 2$, where $|P(i,j)|$ denotes the number of arcs in the path $P(i,j)$.

4.1.4 | Variable upper bounds

Let (i,k) and (k,j) be two arcs traveled by the drone from the launch node i to the customer k and from the customer k to the rendezvous node j, then the variable upper bounds:

$$\omega_i^k \leq x_{ik} \quad (i,k) \in A, \tag{28}$$

and

$$\delta_j^k \leq x_{kj} \quad (k,j) \in A, \tag{29}$$

are valid for the *FS-TSP* formulation. Variable upper bounds (28) and (29) are polynomial in number since they are related to the number of arcs. As a result, these inequalities can be included at the beginning of the formulation.

4.2 | Variable fixing strategies

This subsection presents two families of variable fixings based on the drone endurance. In particular, for each arc $(i,j), (i,j) \in A$, we can set:

$$x_{ij} = 0 \quad (i,j) \in A \ : \ d_{ij} > Dtl - SR, \tag{30}$$

and

$$\gamma_{ij}^k = 0 \quad k \in C, \ (i,j) \in A \ : \ t_{ij} > Dtl - SR, \tag{31}$$

Moreover, if the triangular inequality with respect to the drone travel times and the truck travel times holds and $d_{ij} = d_{ji} \leq t_{ij} = t_{ji}$, for each arc $(i,j), (i,j) \in A$, we can set to zero also other γ_{ij}^k variables with $t_{ij} \leq Dtl - SR$ by considering the following proposition.

Proposition 4. *If $d_{ik} + d_{kj} > 2 \times (Dtl - SR) - t_{ij}$, then $\gamma_{ij}^k = 0$ in any feasible solution.*

Proof. Any sortie where the truck travels the arc (i,j) cannot be in any feasible solution.

Case 1: *Sortie i-k-j.*
If $d_{ik} + d_{kj} + t_{ij} > 2 \times (Dtl - SR)$, then either $d_{ik} + d_{kj} > Dtl - SR$ or $t_{ij} > Dtl - SR$ violate an endurance constraint.

Case 2: *Sortie p-k-q with $p \neq i$ or $q \neq j$.*
If the truck travels the arc (i,j), since the triangular inequality holds, the truck time of the sortie is greater than or equal to $t_{pi} + t_{ij} + t_{jq}$ while the drone time of the sortie is equal to $d_{pk} + d_{kq}$. If $d_{ij} \leq t_{ij}$ for each $(i,j), (i,j) \in A$, the sum of the truck time and the drone time of the sortie $d_{pk} + t_{pi} + t_{ij} + t_{jq} + d_{kq}$ can be decreased by considering the sum $d_{ik} + t_{ij} + d_{kj}$. So, as in **Case 1**, the sortie is not feasible. ∎

4.3 | *B&C* implementation settings

The implementation of the *B&C* is based on a classical framework, envisaging the solution of a relaxed problem and the introduction of violated cuts. Thus, it starts with the solution of the linear programming relaxation obtained considering:

- the set of x, y, γ, θ, δ, and ω variables;
- the subset of truck routing constraints (1) and (2);
- the truck path and launch/rendezvous linking constraints (4)–(6);
- the single assignment constraints (7)–(10);
- the consistency constraints (11)–(16);
- the drone endurance constraints (17) and (18);
- the waiting time constraints (19).

The relaxation is eventually strengthened with the integration of the the valid inequalities (24), (25), (26), (28), and (29), the subset of the valid inequalities (27) with $|P_{ij}| \leq 2$, the variable fixing strategies (30) and (31). Then, at each node of the first three levels of the enumeration tree we separate the valid inequalities (20–23) by using a max flow separation procedure for both integer and fractional solutions [2]. These cuts are separated if the violation exceeds a threshold equal to 0.01. Successively, at each other node of the enumeration tree, if an integer solution is found, we separate the truck routing constraints (3).

5 | COMPUTATIONAL RESULTS

This section presents and discusses the computational results of the experimentation performed to evaluate and validate the proposed formulation and the *B&C* algorithm. The *B&C* algorithm has been coded in C language using Cplex 12.7 with default setting as *MILP* solver, imposing a computation time limit of 1 hour. The experiments are performed on an Intel(R) Core(TM) i7-8700, 3.20 GHz, 16.00 GB of RAM.

The experimentation results are presented in two different subsections. The first subsection shows the impact of valid inequalities and variable fixing strategies on the performance of the *B&C* algorithm. In the second subsection, the performance of the proposed *B&C* algorithm is assessed by comparing it with state-of-the-art solution approaches for the *FS-TSP*.

5.1 | Impact of valid inequalities and variable fixing strategies

In this subsection, we assess the impact of variable fixing strategies and valid inequalities from different perspectives: number of instances optimally solved, gap, running time and quality of the continuous relaxation at the root node. To this end, we tested three different settings:

- *MILP*: this setting involves solving the *MILP* formulation as described in Section 3.
- *MILP-PVI*: this setting involves solving the *MILP* formulation integrated with all the polynomial valid inequalities and variable fixing strategies, that is, constraints (24)–(31). This setting considers all the inequalities that can be added upfront to the model.
- *B&C*: this setting involves solving the *MILP* formulation integrated with valid inequalities and variable fixing strategies. This setting considers all the proposed improvements, including the ones requiring a separation procedure, i.e., constraints (20)–23).

These three settings are tested on the *M10-* and *M20-instances* described in Section 2. We emphasize that we do not consider the *POI-instances* for evaluating the impact of the variable inequalities and variable fixing strategies. This is because, for the *FS-TSP*, the travel time of these instances is rounded up. Therefore, the integer nature of the travel time of these instances can lead to some bias in the results.

5.1.1 | Results on *M10-instances*

The three settings are able to solve all the instances within the time limit. Therefore, Table 2 provides a comparison of the different settings in terms of running times on the *M10-instances*. The instances are grouped based on three different criteria: the position of the depot (Dep), the speed of the drone (s_d), and the drone endurance. For the depot position, the letter "a" indicates that the depot is located near the center of gravity of the customers, while "b," "c," and "d" represent the following (x, y) coordinates: (4.0, 2.7), (4.0, 0.0), and (4, −2.7), respectively. For each instance group, we indicate the number of instances in the column "# Ins." Then, for each setting we report the average and the maximum running time in the columns "Avg time" and "Max time," respectively.

The results demonstrate that all three settings can solve each instance of this test bed within a matter of seconds. Furthermore, it is observed that the running times of the two settings that incorporate the proposed enhancements (*MILP-PVI* and *B&C*) are either comparable or slightly higher than those of the basic setting (*MILP*). This can be attributed to the fact that the formulation associated with the *MILP* setting is already efficient for these small instances, and therefore, the valid inequalities and variable

TABLE 2 Impact of valid inequalities and variable fixing strategies on running times for *M10-instances*.

		MILP		MILP - PVI		B&C	
	# Ins	Avg time	Max time	Avg time	Max time	Avg time	Max time
Dep = a	18	1.52	3.52	2.56	6.58	2.2	4.74
Dep = b	18	1.29	2.60	2.20	4.75	2.00	3.60
Dep = c	18	1.15	2.26	1.63	4.93	1.69	3.30
Dep = d	18	1.24	3.59	1.44	3.14	1.61	3.72
$s_d = 15$	24	1.04	2.13	1.40	3.06	1.55	3.72
$s_d = 25$	24	1.63	3.59	2.28	6.58	2.20	3.42
$s_d = 35$	24	1.23	2.74	2.22	4.93	1.88	4.74
Dtl = 20	36	1.3	3.59	2.41	6.58	1.34	3.42
Dtl = 40	36	1.30	2.39	1.53	4.93	2.42	4.74
All	72	1.30	3.59	1.97	4.93	1.88	4.74

TABLE 3 Impact of valid inequalities and variable fixing strategies on root node relaxation for *M10-instances*.

		MILP		MILP - PVI				B&C			
	# Ins	Avg RGap	Max Rgap	Avg Rgap	Max Rgap	Avg Diff	Max Diff	Avg Rgap	Max Rgap	Avg Diff	Max Diff
Dep = a	18	32.49	40.19	31.86	40.19	0.99	11.95	10.60	21.12	32.90	54.66
Dep = b	18	33.52	43.88	33.01	43.88	0.83	9.13	9.48	24.32	37.23	60.31
Dep = c	18	29.17	42.87	28.38	42.87	1.20	11.42	7.22	24.30	32.10	59.55
Dep = d	18	23.96	35.56	23.07	35.56	1.20	5.49	6.18	16.32	23.96	40.76
$v_d = 15$	24	31.30	42.87	29.97	42.87	2.07	11.95	8.43	24.30	33.99	59.55
$v_d = 25$	24	30.52	43.88	30.09	43.88	0.57	5.47	10.99	24.32	28.96	52.39
$v_d = 35$	24	27.55	37.87	27.18	37.87	0.51	4.66	5.69	14.90	31.69	60.31
Dtl = 20	36	30.59	43.88	29.74	43.88	1.35	11.95	7.51	24.32	34.29	59.87
Dtl = 40	36	28.98	42.87	28.42	42.87	0.75	5.47	9.23	24.30	28.81	60.31
All	72	29.79	43.88	29.08	43.88	1.05	11.95	8.37	24.32	31.55	60.31

fixing strategies only augment the size of the formulation that needs to be solved, without having any other significant impact. However, the *B&C* setting displays superior performance to the *MILP-PVI* setting, demonstrating that the combination of all the enhancements produces better outcomes in terms of running times, even for the small instances, than just the polynomial inequalities.

Furthermore, analyzing the variation of the computation times depending on the instance characteristics, we can observe that all settings have similar trends. Indeed, concerning the depot position, the "a" position, corresponding to the instance center of gravity, is the group of instances that, on average, require more time. This can be explained by considering that a baricentric position of the depot does not address the possible vehicle routes compared to a more specific position. Regarding the drone speed, the most complex instances are those in which the drone has a speed of 25 km/h. It should be noted that for all these instances, the truck speed is always 25 km/h. Therefore, instances where the two vehicles have similar speeds are more complex, as they cannot prefer one vehicle over the other for deliveries. Finally, regarding the last criterion, a higher endurance corresponds to higher computation times since the solution space is larger.

Table 3 presents a comparison of the different settings in terms of continuous relaxation at the root node. Specifically, we solve the corresponding continuous relaxation for each setting to obtain a lower bound for the problem. The percentage gap at the root node *RGap* is then computed as $(\text{BUB} - \text{LB})/\text{LB} \cdot 100$, where *BUB* is the best upper bound known and *LB* is the lower bound obtained by the continuous relaxation. The columns "Avg RGap" and "Max RGap" show the average and maximum *RGap* for each setting. For the *MILP-PVI* and *B&C* settings, we also compute the percentage difference between the *LB* obtained with these settings and the one obtained with the *MILP* setting. This percentage difference is calculated as $(\text{LB} - \text{LB}_{\text{MILP}})/\text{LB}_{\text{MILP}} \cdot 100$, where *LB* is the lower bound computed with the *MILP-PVI* or *B&C* setting, while LB_{MILP} is the lower bound obtained with the *MILP* setting. The average and maximum difference percentage are reported in the columns "Avg Diff" and "Max Diff," respectively.

The results regarding the root node relaxation demonstrate that both the *MILP-PVI* and *B&C* settings lead to an improvement of the lower bound, even on the small instances. Specifically, it can be observed that, on average, the *MILP-PVI* setting improves the lower bound by approximately 1%, with some instances showing improvements of up to 12%. As expected, the *B&C* setting yields even more significant improvements, with an average increase in lower bound of around 31% and a maximum increase of approximately 60%. Furthermore, it is worth noting that, for the *B&C* setting, the average *Rgap* is below 10%, which is lower than the average *Rgap* of the other two settings, and the maximum *Rgap* is even lower than the average *Rgap* of the other settings.

5.1.2 | Results on *M20-instances*

Table 4 presents a comparison of the different settings in terms of number of instances optimally solved, gap, and running times on the *M20-instances*. The instances are grouped based on two criteria: the position of the depot (Dep) and the drone endurance. The letter "C" indicates that the depot is located near the center of gravity of the customers, while "E" and "O" represent the average of the customer x-coordinate with y-coordinate of zero and the southwest corner of the region (0,0), respectively. The column "# Ins" indicates the number of instances in each group. For each setting, we report the number of instances optimally solved within the time limit ("# Opt"), the average and maximum percentage gap ("Avg Gap" and "Max Gap"), and the average running time ("Avg time"). The percentage gap is calculated as $(\text{UB} - \text{LB})/\text{LB} \cdot 100$, where UB and LB represent the upper and lower bounds, respectively, obtained by the different settings.

The results indicate that the running times of the two settings incorporating the proposed enhancements (*MILP-PVI* and *B&C*) are significantly lower than those of the basic setting (*MILP*). Specifically, the average running times of *MILP-PVI* and

TABLE 4 Impact of valid inequalities and variable fixing strategies on number of instances optimally solved, gap, and running times for *M20-instances*.

		MILP				MILP - PVI				B&C			
	# Ins	# Opt	Avg Gap	Max Gap	Avg time	# Opt	Avg Gap	Max Gap	Avg time	# Opt	Avg Gap	Max Gap	Avg time
Dep = C	80	77	0.12	4.15	466.87	79	0.02	1.80	270.73	80	0.00	0.00	190.31
Dep = E	80	72	0.58	14.33	891.21	75	0.36	13.98	544.87	76	0.28	10.30	441.54
Dep = O	80	77	0.12	4.03	541.46	78	0.04	2.19	380.78	79	0.01	0.91	287.49
Dtl = 20	120	117	0.09	4.43	265.58	119	0.05	6.47	93.91	119	0.04	5.31	77.43
Dtl = 40	120	109	0.46	14.33	980.11	113	0.23	13.98	703.68	116	0.15	10.3	535.46
All	240	226	0.27	14.33	622.85	232	0.14	13.98	398.8	235	0.10	10.30	306.44

TABLE 5 Impact of valid inequalities and variable fixing strategies on root node relaxation for *M20-instances*.

		MILP		MILP - PVI				B&C			
	# Ins	Avg RGap	Max Rgap	Avg RGap	Max Rgap	Avg diff	Max diff	Avg RGap	Max Rgap	Avg diff	Max diff
Dep = C	80	34.66	57.81	31.91	54.83	4.38	14.71	14.78	34.93	32.14	97.60
Dep = E	80	44.09	58.61	41.95	56.44	3.95	17.44	22.87	42.91	39.78	93.22
Dep = O	80	40.17	59.22	38.11	57.66	3.71	16.23	19.12	39.07	39.78	96.13
Dtl = 20	120	38.83	58.61	36.73	56.33	3.67	17.44	16.42	42.91	39.06	96.13
Dtl = 40	120	40.45	59.22	37.91	57.66	4.36	15.08	21.43	39.07	34.01	97.60
All	240	39.64	59.22	37.32	57.66	4.02	17.44	18.92	42.91	36.54	97.60

B&C are about 60% and 50% of the average running time of the *MILP* setting, respectively. This is due to the fact that while *MILP* setting can optimally solve 226 out of 240 instances, the *MILP-PVI* and *B&C* settings solve 232 and 235 instances within the time limit, respectively. Furthermore, all the settings have an average gap lower than 0.30%. However, the *MILP-PVI* and *B&C* settings have an average gap that is half and one-third of the *MILP* setting, respectively. Finally, with regard to the maximum percentage gap, it can be observed that *MILP-PVI* yields a slight decrease in the maximum percentage gap compared to the *MILP* setting. On the other hand, the *B&C* setting reduces the maximum percentage gap by about 4%.

Analyzing the variation in the number of instances optimally solved, percentage gap, and computation times depending on the instance characteristics, we can observe that all settings exhibit similar trends on the *M20-instances*. Regarding the depot position, the "E" position, which corresponds to the bottom side of the instance, requires, on average, more time and has the lowest number of instances optimally solved. However, it is worth noting that only eight instances out of 80 are not solved by the *MILP* setting, while the *B&C* setting is unable to solve just four instances of this group. As for the drone endurance, a higher endurance leads to higher computation times and a larger number of instances that are not optimally solved within the time limit, similar to the previous instances. However, unlike the depot position, the drone endurance has a more significant impact on the performance of all settings.

Table 5 compares the different settings in terms of continuous relaxation at the root node for the *M20-instances*. For each setting, we provide the average and maximum percentage root gap, as well as the average and maximum percentage difference.

The root node relaxation results indicate that both the *MILP-PVI* and *B&C* settings provide a significant improvement in the lower bound for the *M20-instances*. On average, the *MILP-PVI* setting leads to a 4% improvement in the lower bound, with some instances showing improvements of up to 18%. Similarly, as observed in the *M10-instances*, the *B&C* setting demonstrates more significant improvements, with an average increase in the lower bound of around 36% and a maximum increase of approximately 97%. It is worth noting that the average *Rgap* for the *B&C* setting is below 20%, which is lower than the average *Rgap* for the other two settings, while the maximum *Rgap* is comparable to the average *Rgap* for the other settings.

5.2 | Comparison with the state-of-the-art solvers for the *FS-TSP*

The results of the comparison with state-of-the-art solution approaches are presented in the following three sub-sections, one for each of the test beds described in Section 2. Specifically, the performance of the proposed *B&C* method is evaluated by comparing it with the state-of-the-art solution approaches for each test bed. We would like to note that the results will be presented in an aggregated form for the sake of readability. Nevertheless, detailed results can be found in the Appendix for those interested.

TABLE 6 Comparison of the speed-up with respect to *DMN22* and *BMS21*.

		Speed-up B&C versus DMN21				Speed-up B&C versus BMS21			
	# Ins	# Faster	Average	Min	Max	# Faster	Average	Min	Max
Dep = a	18	18	640.46	1.28	6448.85	18	190.39	2.03	1254.99
Dep = b	18	18	119.43	1.10	691.19	18	71.28	1.64	383.96
Dep = c	18	17	33.01	0.48	172.50	18	40.20	1.82	260.01
Dep = d	18	18	28.71	1.57	102.26	17	42.36	0.68	220.85
$s_d = 15$	24	24	153.34	1.10	1240.90	23	166.44	0.68	1254.99
$s_d = 25$	24	23	357.05	0.48	6448.85	24	69.63	2.58	592.12
$s_d = 35$	24	24	105.82	1.74	1112.39	24	22.10	1.43	108.46
Dtl = 20	36	36	36.81	1.10	287.46	35	24.44	0.68	195.81
Dtl = 40	36	35	374.00	0.48	6448.85	36	147.68	1.43	1254.99
All	72	71	205.40	0.48	6448.85	71	86.06	0.68	1254.99

The state-of-the-art solution approaches for the *M10-instances* and the *M20-instances* are represented by the Column-and-Row Generation method proposed in [4] and the Branch-and-Cut method proposed in [11], which we will refer to as *BMS21* and *DMN22*, respectively. On the other hand, the state-of-the-art approach for the *FS-TSP* on the *POI-instances* is represented by the solution method proposed in [22], which we will refer to as *RR21*.

It is important to note that the results related to *BMS21*, *DMN22*, and *RR21* are taken from [4], [11], and [22], respectively. Therefore, a direct comparison in terms of running time is not possible since the different solution methods were tested on different computing environments. However, it is worth mentioning that the CPU used in this work is similar to the one used in *RR21*, and slightly better than those used in *DMN22* and *BMS21*, as indicated by a study reported in [8]. This study ranks processors based on single-thread performance, with a range of scores from 77 to 126 045. In particular, the marks of the CPU used for the *B&C*, *DMN22*, *BMS21*, and *RR21* are 12958, 1843, 3275, and 11490, respectively.

5.2.1 | Results on *M10-instances*

All the methods considered for the *M10-instances* were able to obtain the optimal solution for all the instances in this set. Thus, the only way to compare their performance is in terms of running times. To this end, we computed the speed-up related to the *B&C* approach in comparison to *DMN22* and *BMS21*. The speed-up is computed by calculating the ratio of the running time of the literature method and our running time. A speed-up lower than 1 indicates that the *B&C* approach is slower than the literature method, while a speed-up greater than 1 means that it results in the same running time or is faster.

Table 6 reports the results of this comparison. We grouped the instances based on the depot location, drone speed, and drone endurance. For each group, we indicate the number of instances ("# Ins"). Then, for each literature method, we indicate the number of instances that our *B&C* approach solves faster ("# Faster"), the average, the minimum, and the maximum speed-up achieved.

The results show that the *B&C* is significantly faster than the two literature approaches. In fact, we can observe that on average with respect to *DMN22*, the *B&C* shows a speed-up of about 200. Furthermore, we can also observe that the *B&C* is slower than *DMN22* only on one instance, where *DMN22* is twice as fast. However, on the instance where *DMN22* performs worse, the *B&C* is about 6000 times faster. Similar considerations can be made for the comparison with *BMS21*. In summary, we can observe that the average speed-up is about 80 while the maximum speed-up is about 1200. In conclusion, although a direct comparison is not possible due to the different environments used for the tests, the speed-up values are so high that they cannot be justified only by the different computing power involved, demonstrating the effectiveness of the proposed solution method.

5.2.2 | Results on *M20-instances*

The state-of-the-art solution approaches are not capable of obtaining the optimal solution for all the *M20-instances* within the given time limit of 1 h. Furthermore, the running times of *DMN22* are not available for this set of instances, making it impossible to compare the running times of the three solution methods. Hence, we assess the methods based on the percentage optimality gap. Specifically, in Table 7, we provide the number of optimal solutions achieved, along with the average and maximum percentage gap of the two literature approaches. For each instance, we select the best result among *DMN22* and *BMS21*. To avoid redundancy, we do not report again the gap for our *B&C* present the number of instances where our *B&C* approach exhibits a higher, equal, or lower gap in the columns "# Worse," "# Equal," "# Better," respectively.

TABLE 7 Comparison of quality of the solution with respect to *DMN22* and *BMS21*.

		DMN22/BMS21			B&C versus DMN22/BMS21		
	# Ins	# Opt	Avg Gap	Max Gap	# Worse	# Equal	# Better
Dep = C	80	27	10.42	39.00	0	27	53
Dep = E	80	20	11.24	45.63	1	20	59
Dep = O	80	41	10.15	49.66	0	41	39
Dtl = 20	120	82	2.54	23.99	1	82	37
Dtl = 40	120	6	18.67	49.66	0	6	114
All	240	88	10.60	49.66	1	88	151

TABLE 8 Comparison with *RR21* in terms of number of instances optimally solved, gap, and running time.

		RR21				B&C			
	#Ins	# Opt	Avg Gap	Max Gap	Avg time	# Opt	Avg gap	Max gap	Avg time
|C| = 9	75	75	0.00	0.00	0.13	75	0.00	0.00	0.18
|C| = 19	75	75	0.00	0.00	3.18	75	0.00	0.00	6.07
|C| = 29	75	75	0.00	0.00	92.29	74	0.03	2.38	198.94
|C| = 39	75	61	5.48	51.01	1475.88	57	0.58	6.22	1482.69
$\alpha = 1$	100	94	0.88	33.57	405.01	97	0.05	1.85	238.75
$\alpha = 2$	100	98	0.82	43.51	279.89	94	0.16	6.22	424.72
$\alpha = 3$	100	94	2.42	51.01	493.70	90	0.25	5.43	602.43
All	300	286	1.37	51.01	392.87	281	0.15	6.22	421.97

The results show how, even considering the best result between the two state-of-the-art approaches, the results of our *B&C* approach are significantly better. In fact, we can observe that the average percentage gap is 10.60, which is higher than the maximum gap of our *B&C* approach. Moreover, by comparing the instances where the *B&C* approach obtains a lower gap, it can be noticed that in 151 out of 240 instances, our *B&C* approach outperforms the literature approaches, which is about 60% of the instances. Furthermore, it can be observed that only in one instance, the literature approaches perform better than our *B&C* approach. Finally, considering the different characteristics of the instances, it can be observed that the main improvement is related to the ability of the *B&C* approach to solve instances with a drone endurance of 40. In fact, in 114 out of 120 instances, the *B&C* approach obtains a lower gap.

5.2.3 | Results on *POI-instances*

The *POI-instances* can be grouped on the basis of two different features: the number of customer and the ratio between the drone and the truck speed (α). Table 8 shows the comparison between *RR21* and the *B&C* on the different subsets of *POI-instances*. For each subset, we report the number of instances. Then, for each solution method we report the number of instances solved to optimality within the time limit, the average and the maximum percentage gap, and the average running time.

The performance comparison of the two methods based on instance characteristics reveals that their performance tends to decrease with an increase in the number of clients and drone speed. In the former case, the instance size increases, while in the latter case, the solution space grows as more clients can be served by the drone since the endurance is fixed at 20 for all speeds. Furthermore, the results demonstrate that the two approaches yield similar results in terms of the number of optimal solutions. Specifically, *RR21* solves 286 out of 300 instances, while the *B&C* solves 281. However, the *B&C* generally produces significantly lower gaps than *RR21* in terms of percentage. In particular, the average and maximum gaps are 0.15% and 6.22%, respectively, for the *B&C*, while they are 1.37% and 51.01% for *RR21*.

Based on these results, we can conclude that the proposed *B&C* is more robust than *RR21*. Although the *B&C* solves, on average, slightly fewer instances to optimality than *RR21*, the instances that the *B&C* is unable to solve have significantly lower gaps compared to *RR21*, which presents particularly high gaps for unsolved instances. To further support this conclusion, we reported a detailed comparison on the unsolved instances by *RR21* and *B&C* in Tables 9 and 10, respectively.

From Table 9, we can observe that, on the instances unsolved by *RR21*, its average gap is equal to 29.37% while our *B&C* determines an average gap of 0.84%. On the other hand, as shown in Table 10, on the instances unsolved by the *B&C*, its average gap is 2.40% while the one of *RR21* is 10.53%. Therefore, on average, our *B&C* performs significantly better than *RR21* on both subsets of unsolved instances.

TABLE 9 Results on *POI-instances* unsolved by *RR21*.

	RR21		*B&C*	
Id	**UB**	**% Gap**	**UB**	**%Gap**
poi-40-11-1	266	0.75	266	0.00
poi-40-13-3	363	42.15	210	0.95
poi-40-15-3	346	33.53	231	4.33
poi-40-17-3	359	32.59	242	0.00
poi-40-18-1	391	25.32	297	1.68
poi-40-22-3	339	34.22	224	0.00
poi-40-23-1	395	26.84	302	0.00
poi-40-23-2	395	38.48	244	0.00
poi-40-25-1	417	33.57	281	0.00
poi-40-3-2	416	43.51	235	0.00
poi-40-3-3	416	48.08	216	3.24
poi-40-4-1	284	0.35	288	0.00
poi-40-7-1	271	0.74	271	0.00
poi-40-7-3	396	51.01	194	1.55

TABLE 10 Results on *POI-instances* unsolved by *B&C*.

	RR21		*B&C*	
Id	**UB**	**% Gap**	**UB**	**%Gap**
poi-30-20-3	168	0.00	168	2.38
poi-40-1-2	213	0.00	214	0.93
poi-40-13-3	363	42.15	210	0.95
poi-40-15-1	279	0.00	279	1.43
poi-40-15-3	346	33.53	231	4.33
poi-40-16-1	270	0.00	270	1.85
poi-40-16-2	210	0.00	210	0.95
poi-40-18-1	391	25.32	297	1.68
poi-40-18-2	239	0.00	241	6.22
poi-40-18-3	221	0.00	221	5.43
poi-40-21-3	218	0.00	218	1.38
poi-40-24-2	215	0.00	215	2.33
poi-40-25-3	190	0.00	190	1.05
poi-40-3-3	416	48.08	216	3.24
poi-40-4-2	234	0.00	234	1.71
poi-40-5-3	203	0.00	205	3.41
poi-40-6-2	192	0.00	192	3.65
poi-40-6-3	181	0.00	181	1.10
poi-40-7-3	396	51.01	194	1.55

6 | CONCLUSIONS

In this study, we propose an original *MILP* formulation for the *FS-TSP*, which unlike the ones present in the literature, uses a polynomial number of variables and does not use *big-M* constraints.

We solve the proposed formulation through a *B&C* approach integrated with drone endurance based variable fixing strategies and four different families of valid inequalities based on the vehicle coordination.

Computational results on different sets of benchmark instances confirm the robustness of the proposed *MILP* formulation and the effectiveness of the *B&C*. Indeed, the experimentation shows that the proposed method is competitive or outperforms the state-of-the-art approaches, providing either the optimal solution or improved bounds for several instances unsolved before.

Future research directions naturally include the extension of the proposed formulation to different variants of the *FS-TSP*. Moreover, we are interested in developing a matheuristic framework that includes the proposed formulation to solve larger and

harder instances. Finally, we will investigate the possibility of combining the proposed solution approach with machine learning and data science techniques to be able to compute good solutions with a limited computational burden.

FUNDING INFORMATION

This research was partially supported by the Italian University and Research Ministry, within the activities of project "Centro Nazionale per la Mobilità Sostenibile (CNMS)", CUP E63C22000930007, CN00000023, Spoke 10 - Logistics, developed at University Federico II of Naples.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] P. Avella, M. Boccia, and A. Sforza, *A branch-and-cut algorithm for the median-path problem*, Comput. Optim. Appl. **32** (2005), 215–230.
[2] M. Boccia, T. G. Crainic, A. Sforza, and C. Sterle, *Multi-commodity location-routing: Flow intercepting formulation and branch-and-cut algorithm*, Comput. Oper. Res. **89** (2018), 94–112.
[3] M. Boccia, A. Mancuso, A. Masone, and C. Sterle. *A feature based solution approach for the flying sidekick traveling salesman problem*, Int. Conf. Math. Optim. Theory Oper. Res. 2021, pp. 131–146.
[4] M. Boccia, A. Masone, A. Sforza, and C. Sterle, *A column-and-row generation approach for the flying sidekick travelling salesman problem*, Transp. Res. Part C: Emerg. Technol. **124** (2021), 102913.
[5] M. Boccia, A. Masone, A. Sforza, and C. Sterle, *An exact approach for a variant of the fs-tsp*, Transp. Res. Proc. **52** (2021), 51–58.
[6] S. Cavani, M. Iori, and R. Roberti, *Exact methods for the traveling salesman problem with multiple drones*, Transp. Res. Pt C. Emerg. Technol. **130** (2021), 103280.
[7] I. M. Chao, *A tabu search method for the truck and trailer routing problem*, Comput. Oper. Res. **29** (2002), 33–51.
[8] CPU Benchmarks. PassMark Software, April 4 2023.
[9] M. Dell'Amico, R. Montemanni, and S. Novellani, *Algorithms based on branch and bound for the flying sidekick traveling salesman problem*, Omega **104** (2021), 102493.
[10] M. Dell'Amico, R. Montemanni, and S. Novellani, *Drone-assisted deliveries: New formulations for the flying sidekick traveling salesman problem*, Optim. Lett. **15** (2021), 1617–1648.
[11] M. Dell'Amico, R. Montemanni, and S. Novellani, *Exact models for the flying sidekick traveling salesman problem*, Int. Trans. Oper. Res. **29** (2022), 1360–1393.
[12] J. C. de Freitas and P. H. V. Penna, *A variable neighborhood search for flying sidekick traveling salesman problem*, Int. Trans. Oper. Res. **27** (2020), 267–290.
[13] A. Gunawan, H. C. Lau, and P. Vansteenwegen, *Orienteering problem: A survey of recent variants, solution approaches and applications*, Eur. J. Oper. Res. **255** (2016), 315–332.
[14] D. R. Karger and C. Stein, *A new approach to the minimum cut problem*, J. ACM **43** (1996), 601–640.
[15] G. Macrina, L. Di Puglia Pugliese, F. Guerriero, and G. Laporte, *Drone-aided routing: A literature review*, Transp. Res. Emerg. Technol. Pt. C **120** (2020), 102762.
[16] A. Masone, S. Poikonen, and B. L. Golden, *The multivisit drone routing problem with edge launches: An iterative approach with discrete and continuous improvements*, Networks **80** (2022), 193–215.
[17] C. C. Murray and A. G. Chu, *The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery*, Transp. Res. Pt. C Emerg. Technol. **54** (2015), 86–109.
[18] C. C. Murray and R. Raj, *The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones*, Transp. Res. Pt. C Emerg. Technol. **110** (2020), 368–398.
[19] W. Najy, C. Archetti, and A. Diabat, *Collaborative truck-and-drone delivery for inventory-routing problems*, Transp. Res. Pt. C Emerg. Technol. **146** (2023), 103791.
[20] S. Poikonen and B. Golden, *Multi-visit drone routing problem*, Comput. Oper. Res. **113** (2020), 104802.
[21] S. Poikonen, B. Golden, and E. A. Wasil, *A branch-and-bound approach to the traveling salesman problem with a drone*, INFORMS J Com. **31** (2019), 335–346.
[22] R. Roberti and M. Ruthmair, *Exact methods for the traveling salesman problem with drone*, Transp. Sci. **55** (2021), 275–552.
[23] D. Rojas Viloria, E. L. Solano-Charris, A. Muñoz-Villamizar, and J. R. Montoya-Torres, *Unmanned aerial vehicles/drones in vehicle routing problems: A literature review*, Int. Trans. Oper. Res. **28** (2020), 1626–1657.
[24] D. Schermer, M. Moeini, and O. Wendt, *A matheuristic for the vehicle routing problem with drones and its variants*, Transp. Res. Pt C. Emerg. Technol. **106** (2019), 166–204.
[25] D. Schermer, M. Moeini, and O. Wendt, *A branch-and-cut approach and alternative formulations for the traveling salesman problem with drone*, Networks **76** (2020), 164–186.
[26] Y. Wang, Z. Wang, X. Hu, G. Xue, and X. Guan, *Truck–drone hybrid routing problem with time-dependent road travel time*, Transp. Res. Pt C. Emerg. Technol. **144** (2022), 103901.
[27] E. Yurek and H. Ozmutlu, *A decomposition-based iterative optimization algorithm for traveling salesman problem with drone*, Transp. Res. Pt C. Emerg. Technol. **91** (2018), 249–262.

How to cite this article: M. Boccia, A. Mancuso, A. Masone, and C. Sterle, *A new MILP formulation for the flying sidekick traveling salesman problem*, Networks. **82** (2023), 254–276. https://doi.org/10.1002/net.22172

APPENDIX

This appendix contains the detailed results of the comparison of the proposed *B&C* with the state-of-the-art approaches for the *FS-TSP*.

A.1 | M10-instances

Tables A1 and A2 report the results for the instances with a *Dtl* equal to 20 and 40 min, respectively. For each instance, we report the objective function value of the optimal solution and the running times of each solution method.

Concerning the results on the instances with *Dtl* equal to 20 min, we can observe that the proposed *B&C* is faster than the other two methods. Indeed, it shows an average computation Time of 1.34 seconds. On the other hand, the other two approaches are more than 10 times slower than the proposed one, with an average computation time equal to 45.27 and 75.56 s for *BMS21* and *DMN22*, respectively. Moreover, our approach performs better than the others, even in the worst-case scenario, that is, by considering the largest computation time required to solve an instance. In particular, the largest computation time for the *B&C* is 3.42 s versus the 329.71 s for *BMS21* and the 707.32 s for *DMN22*.

Concerning, instead, the instances with *Dtl* equal to 40 min, we can observe that the performance of the *B&C* seems to be unaffected by the larger *Dtl*. Indeed, the average and the largest computation times are similar to the previous ones. In particular, the average and the largest computation times are slightly larger (2.42 and 4.74 s, respectively). Instead, the performance of the other two solution methods decreases when the *Dtl* increases. Indeed, on the one hand, the average and the largest computation times for *BMS21* are 361.08 and 2762.93 s, respectively. On the other hand, *DMN22* shows an average computation time of 975.44 and a largest computation time of about 4 hours. In conclusion, the *B&C* outperforms the state-of-the-art algorithms on the *M10-instances*, being faster than the other approaches by about a couple of orders of magnitude.

TABLE A1 Results on *M10-instances* instances with *Dtl* = 20.

		DMN22	BMS21	B&C			DMN22	BMS21	B&C			DMN22	BMS21	B&C
Id	OPT	Time	Time	Time	Id	OPT	time	time	time	Id	OPT	time	time	time
M37V1	57.45	1.23	3.80	0.96	M40V1	49.43	4.03	12.14	0.81	M43V1	69.59	0.53	0.3	0.15
M37V2	53.79	0.79	1.96	0.72	M40V2	51.71	12.09	27.93	0.84	M43V2	72.15	0.58	0.3	0.18
M37V3	54.66	6.21	2.78	0.41	M40V3	57.1	6.9	9.11	0.91	M43V3	77.34	0.43	0.24	0.12
M37V4	67.46	5.09	1.03	0.33	M40V4	69.9	1.98	11.26	0.80	M43V4	90.14	0.53	0.23	0.34
M37V5	51.78	406.43	190.25	3.42	M40V5	45.46	132.01	70.81	2.45	M43V5	58.71	282.87	192.68	0.98
M37V6	48.6	128.31	80.29	2.65	M40V6	44.51	9.48	35.44	1.31	M43V6	59.09	69.65	55.03	1.54
M37V7	49.58	18.38	32.77	2.36	M40V7	49.9	5.27	7.78	1.15	M43V7	65.52	8.98	7.7	1.24
M37V8	62.38	43.12	35.87	1.99	M40V8	62.7	5.84	8.97	0.93	M43V8	84.81	103.32	20.98	1.66
M37V9	43.48	361.49	329.71	3.04	M40V9	42.53	14.47	9.57	1.25	M43V9	46.93	707.32	239.28	2.94
M37V10	41.91	211.62	170.37	2.03	M40V10	43.08	3.64	3.78	2.05	M43V10	47.93	72.36	51.35	2.69
M37V11	42.9	37.47	16.02	1.45	M40V11	49.2	1.19	1.26	0.68	M43V11	57.38	11.91	5.52	0.85
M37V12	56.85	38.31	25.85	1.44	M40V12	62	1.12	1.13	0.62	M43V12	69.2	5.28	2.09	0.78

TABLE A2 Results on *M10-instances* with *Dtl* = 40.

		DMN22	BMS21	B&C			DMN22	BMS21	B&C			DMN22	BMS21	B&C
Id	OPT	Time	Time	Time	Id	OPT	Time	Time	Time	Id	OPT	Time	Time	Time
M37V1	50.50	2731.91	2762.93	2.20	M40V1	46.89	252.66	365.46	3.16	M43V1	57.01	2312.62	1643.65	2.35
M37V2	47.31	284.46	550.88	1.43	M40V2	46.42	70.15	123.30	2.31	M43V2	58.05	1446.23	792.02	2.24
M37V3	53.69	190.49	618.77	2.38	M40V3	53.93	150.67	502.94	3.20	M43V3	69.40	219.80	222.88	2.33
M37V4	67.46	148.43	568.48	2.57	M40V4	68.40	252.36	608.64	3.72	M43V4	83.70	224.88	397.32	2.82
M37V5	45.84	644.23	369.05	2.75	M40V5	43.53	53.75	48.96	2.62	M43V5	52.09	15702.75	1441.79	2.43
M37V6	44.60	253.15	178.55	3.17	M40V6	44.08	40.61	13.23	1.92	M43V6	52.33	1921.06	374.47	2.78
M37V7	47.62	267.10	349.93	3.30	M40V7	49.23	1.13	6.05	2.35	M43V7	61.88	452.45	77.90	2.62
M37V8	60.42	241.61	299.12	2.36	M40V8	62.03	6.76	5.62	1.83	M43V8	73.73	123.31	56.58	2.86
M37V9	42.42	1167.62	62.31	2.08	M40V9	42.53	5.50	7.63	1.25	M43V9	46.93	5271.24	362.06	4.74
M37V10	41.91	406.15	76.01	2.22	M40V10	43.08	5.53	4.94	2.34	M43V10	47.93	169.56	82.85	3.61
M37V11	42.90	51.09	9.15	2.44	M40V11	49.20	3.38	1.70	0.80	M43V11	56.40	11.37	3.30	1.82
M37V12	55.70	26.62	6.69	2.20	M40V12	62.00	1.16	0.88	0.62	M43V12	69.20	4.21	2.94	1.17

A.2 | M20-instances

The two literature algorithms are not able to solve to optimality several instances even with *Dtl* equal to 20. Therefore, we report the best upper bound and best percentage optimality gap among the two methods for each instance. Then, we report the upper bound, the percentage gap, and the computation time obtained by our *B&C*. The percentage optimality gap is calculated as $\%Gap = (\mathrm{UB} - \mathrm{LB})/\mathrm{LB} \cdot 100$. We highlight that we do not report a detailed comparison of the running times of the different solution methods since the running times are not available in [11] for *DMN22*.

The results on the *M20-instances* with *Dtl* equal to 20 and 40 min are reported in Tables A3 and A4, respectively.

As shown in Table A3, the two literature approaches can solve only 82 out of 120 instances to optimality. Instead, our approach solves to optimality almost all the instances (119 out of 120). Clearly, our approach shows better performance also in terms of average and maximum optimality gap. Indeed, our optimality gap is equal to 0.04% versus the 2.54% of *BMS21*/*DMN22*.

TABLE A3 Results on *M20-instances* with *Dtl* = 20.

	DMN22/BMS21						*DMN22/BMS21*						*DMN22/BMS21*					
	Best	Best	*B&C*				Best	Best	*B&C*				Best	Best	*B&C*			
Id	UB	% Gap	UB	%Gap	Time	Id	UB	% Gap	UB	% Gap	Time	Id	UB	% Gap	UB	%Gap	Time
4847	267.05	0.00	267.05	0.00	5.39	5025	131.43	0.00	131.43	0.00	9.87	5154	123.34	9.84	121.90	0.00	49.95
4849	248.30	0.00	248.30	0.00	0.84	5027	114.31	0.00	114.31	0.00	14.32	5156	124.46	0.00	124.46	0.00	21.18
4853	232.87	0.00	232.87	0.00	1.01	5030	116.1	0.00	116.10	0.00	44.69	5159	145.79	0.00	145.79	0.00	18.46
4856	253.33	0.00	253.33	0.00	1.47	5032	117.55	1.88	117.55	0.00	13.28	5201	148.02	0.00	148.02	0.00	9.25
4858	240.63	0.00	240.63	0.00	1.39	5034	105.10	1.18	105.10	0.00	42.98	5203	138.59	0.00	138.59	0.00	21.24
4902	242.32	0.00	242.32	0.00	1.25	5036	124.33	3.65	124.33	0.00	22.99	5205	134.78	0.00	134.78	0.00	49.77
4907	239.28	0.00	239.28	0.00	0.91	5039	130.91	3.67	130.91	0.00	7.50	5207	121.47	0.00	121.47	0.00	23.18
4909	222.88	0.00	222.88	0.00	1.15	5041	125.32	7.77	125.32	0.00	10.83	5209	135.92	0.00	135.92	0.00	15.72
4912	267.62	0.00	267.62	0.00	0.87	5044	121.87	11.51	120.77	0.00	25.80	5212	137.67	0.00	137.67	0.00	18.77
4915	259.39	0.00	259.39	0.00	0.66	5047	112.85	0.00	112.85	0.00	9.61	5214	126.25	0.00	126.25	0.00	27.93
4917	173.97	0.00	173.97	0.00	3.60	5049	197.66	0.00	197.76	0.00	3.31	5216	101.07	18.27	101.07	0.00	46.56
4920	170.05	0.00	170.05	0.00	4.92	5051	180.62	0.00	180.62	0.00	9.92	5218	115.82	4.14	115.62	0.00	27.93
4922	169.60	0.00	169.60	0.00	2.66	5053	176.51	0.00	176.51	0.00	2.01	5220	119.04	0.00	119.04	0.00	28.49
4924	159.57	0.00	159.57	0.00	2.56	5055	177.29	0.00	177.29	0.00	4.23	5223	94.59	3.64	94.59	0.00	42.48
4926	155.98	0.00	155.98	0.00	13.54	5057	180.7	0.00	180.70	0.00	6.02	5225	129.72	0.00	129.72	0.00	12.36
4928	166	0.00	166.00	0.00	18.64	5059	150.82	0.00	150.82	0.00	5.64	5227	116.19	0.00	116.19	0.00	75.77
4931	172.49	0.00	172.49	0.00	14.91	5102	165.49	0.00	165.49	0.00	5.24	5229	94.26	13.69	92.87	0.00	70.17
4933	159.39	0.00	159.39	0.00	13.83	5104	181.61	0.00	181.61	0.00	4.93	5231	98.93	0.00	98.36	0.00	38.46
4935	176.69	0.00	176.69	0.00	18.86	5106	158.49	0.00	158.49	0.00	12.78	5233	111.62	7.85	111.62	0.00	29.04
4937	173.55	0.00	172.50	0.00	5.51	5108	172.12	0.00	172.12	0.00	1.96	5235	118.89	0.00	118.89	0.00	12.20
4939	201.03	0.00	201.03	0.00	5.34	5110	135.43	0.00	135.43	0.00	17.93	5238	79.39	10.33	78.93	0.00	562.17
4941	253.08	0.00	253.08	0.00	1.27	5112	131.23	0.00	131.23	0.00	13.20	5240	87.46	11.84	84.51	0.00	183.79
4944	247.03	0.00	247.03	0.00	2.98	5115	127.39	0.00	127.39	0.00	19.19	5242	85.65	3.25	85.65	0.00	10.29
4946	237.21	0.00	237.21	0.00	1.36	5117	130.36	10.31	130.36	0.00	46.49	5244	86.81	0.00	86.81	0.00	33.92
4948	258.06	0.00	258.06	0.00	1.38	5119	118.97	0.00	118.97	0.00	28.65	5246	74.56	5.12	74.19	5.31	3600
4950	240.99	0.00	240.99	0.00	1.44	5121	131.84	8.90	131.19	0.00	25.25	5248	83.1	23.36	83.10	0.00	1760.19
4952	218.09	0.00	218.09	0.00	4.77	5123	121.95	1.61	121.95	0.00	16.46	5250	81.93	0.00	81.93	0.00	42.83
4954	261.06	0.00	261.06	0.00	1.40	5125	130.96	7.38	130.96	0.00	13.52	5252	86.38	2.42	86.15	0.00	110.90
4957	252.28	0.00	252.28	0.00	1.20	5127	132.24	0.00	132.24	0.00	7.97	5255	82.5	2.83	82.50	0.00	89.69
4959	249.92	0.00	249.92	0.00	1.11	5130	126.49	9.37	126.49	0.00	45.19	5257	79.43	23.99	78.27	0.00	329.14
5001	115.5	0.00	115.50	0.00	17.58	5132	106.36	14.92	106.36	0.00	50.90	5306	100.46	0.00	100.46	0.00	19.14
5003	173.54	3.47	172.73	0.00	8.03	5134	102.57	0.76	102.57	0.00	21.86	5310	92.41	0.00	92.41	0.00	47.10
5006	155.39	0.00	155.39	0.00	5.90	5136	98.91	0.00	98.91	0.00	32.02	5312	83.59	0.00	83.59	0.00	20.19
5008	159.74	0.00	159.74	0.00	5.87	5138	91.83	3.81	91.61	0.00	73.10	5321	101.49	0.00	101.49	0.00	11.89
5010	145.48	0.00	145.48	0.00	15.00	5141	95.53	16.82	95.53	0.00	68.49	5324	104.03	8.98	101.55	0.00	77.96
5012	172.4	0.00	171.38	0.00	8.02	5143	97.69	0.00	97.69	0.00	26.96	5330	118.45	0.00	118.45	0.00	7.55
5015	172.67	0.00	172.67	0.00	3.43	5145	95.09	5.89	95.09	0.00	43.67	5334	102.02	0.00	102.02	0.00	26.40
5017	166.47	7.13	166.47	0.00	8.86	5148	90.58	7.12	90.58	0.00	52.35	5336	104.46	0.00	104.46	0.00	12.53
5020	155.92	0.00	155.92	0.00	3.46	5150	82.19	9.99	82.19	0.00	172.37	5345	114.19	0.00	114.19	0.00	7.88
5022	146.21	3.41	146.21	0.00	13.71	5152	90.58	14.38	90.58	0.00	384.83	5351	115.21	0.44	115.10	0.00	53.96

TABLE A4 Results on *M20-instances* with *Dtl* = 40.

	DMN22/BMS21						*DMN22/BMS21*						*DMN22/BMS21*				
	Best	**Best**	***B&C***				**Best**	**Best**	***B&C***				**Best**	**Best**	***B&C***		
Id	**UB**	**% Gap**	**UB**	**%Gap**	**Time**	**Id**	**UB**	**% Gap**	**UB**	**% Gap**	**Time**	**Id**	**UB**	**% Gap**	**UB**	**%Gap**	**Time**
4847	255.60	10.70	255.60	0.00	51.14	5025	119.20	32.26	118.43	0.00	533.62	5154	106.64	20.26	104.28	0.00	296.76
4849	225.15	22.37	225.15	0.00	108.81	5027	112.15	11.51	111.81	0.00	333.10	5156	108.92	16.20	107.12	0.00	120.46
4853	218.60	18.68	211.38	0.00	115.55	5030	102.69	10.35	100.64	0.00	3490.25	5159	125.41	17.76	120.02	0.00	158.89
4856	237.03	21.12	234.63	0.00	113.71	5032	106.69	12.01	103.06	0.00	204.23	5201	140.30	19.87	140.30	0.00	813.36
4858	215.63	13.81	215.63	0.00	52.83	5034	102.63	0.71	102.16	0.00	168.37	5203	124.20	16.32	124.06	0.00	359.09
4902	226.62	23.27	223.84	0.00	48.92	5036	112.30	1.06	112.10	0.00	88.98	5205	119.79	40.06	117.68	0.00	329.91
4907	196.66	17.33	192.87	0.00	41.95	5039	126.96	35.76	115.85	10.30	3600	5207	113.90	30.37	113.90	0.00	318.79
4909	216.04	7.95	216.04	0.00	48.41	5041	114.55	17.12	113.88	0.00	276.09	5209	123.69	49.66	121.24	0.91	3600
4912	238.04	1.83	238.04	0.00	51.64	5044	115.94	15.51	115.09	0.00	510.86	5212	135.82	22.84	133.06	0.00	443.64
4915	234.42	13.26	229.03	0.00	39.29	5047	106.11	20.30	103.86	0.00	354.80	5214	123.03	42.11	121.85	0.00	282.34
4917	165.31	2.17	165.31	0.00	29.56	5049	183.51	20.39	180.71	0.00	1065.02	5216	93.31	14.51	91.94	0.00	146.80
4920	161.89	6.22	157.03	0.00	131.76	5051	165.69	15.32	162.52	0.00	803.47	5218	97.61	21.93	96.06	0.00	161.47
4922	168.10	14.46	164.10	0.00	133.30	5053	145.54	26.26	145.42	0.00	115.44	5220	118.79	16.38	114.93	0.00	171.70
4924	158.98	3.92	158.98	0.00	93.41	5055	172.30	9.87	171.64	0.00	77.22	5223	93.64	22.49	91.22	0.00	503.58
4926	152.24	45.63	151.19	0.00	305.57	5057	173.67	10.18	172.88	0.00	479.58	5225	126.57	7.85	123.58	0.00	84.26
4928	153.88	40.19	146.76	5.00	3600	5059	134.83	14.06	134.83	0.00	261.99	5227	105.30	18.49	101.74	0.00	149.88
4931	166.68	11.64	165.22	0.00	162.10	5102	164.01	19.12	163.77	0.00	124.13	5229	94.26	17.15	88.67	0.00	70.36
4933	157.19	25.17	155.13	0.00	401.80	5104	178.67	17.80	176.70	0.00	104.39	5231	99.17	29.33	94.02	0.00	1583.46
4935	160.97	22.86	155.77	0.00	1860.91	5106	144.37	33.42	141.77	0.00	196.89	5233	108.03	16.81	107.9	0.00	138.11
4937	159.44	8.93	159.18	0.00	490.75	5108	162.21	10.00	162.21	0.00	935.15	5235	98.96	19.80	98.96	0.00	357.60
4939	180.92	14.17	179.10	0.00	2986.07	5110	129.07	22.71	126.20	0.00	1322.61	5238	79.09	17.67	77.99	0.00	281.27
4941	241.90	4.71	241.90	0.00	26.95	5112	129.56	24.30	123.13	0.00	287.31	5240	83.27	31.17	81.02	0.00	134.01
4944	233.75	0.00	233.75	0.00	54.47	5115	126.90	21.09	123.08	0.00	3115.85	5242	85.65	6.77	84.52	0.00	81.70
4946	220.39	0.00	220.39	0.00	75.87	5117	124.10	27.64	118.87	0.00	505.66	5244	85.55	36.27	85.32	0.00	33.78
4948	244.13	0.91	244.13	0.00	18.35	5119	114.18	28.82	112.03	0.00	223.39	5246	63.74	18.28	63.25	0.00	125.96
4950	225.61	0.00	225.61	0.00	28.95	5121	118.22	7.72	118.22	0.00	144.85	5248	83.97	39.64	77.55	0.00	199.20
4952	211.67	6.97	210.67	0.00	346.84	5123	115.40	18.73	113.75	0.00	229.14	5250	81.73	24.78	81.73	0.00	2120.91
4954	240.59	9.46	239.67	0.00	40.83	5125	120.17	25.42	118.86	0.00	592.08	5252	85.33	15.15	85.33	1.94	3600
4957	231.61	9.78	225.85	0.00	88.06	5127	125.95	13.07	124.57	0.00	219.70	5255	72.76	27.81	71.13	0.00	345.29
4959	231.95	0.00	231.95	0.00	36.42	5130	120.53	25.07	119.01	0.00	405.15	5257	79.35	27.36	75.17	0.00	753.53
5001	114.09	4.37	114.09	0.00	129.04	5132	107.47	25.96	102.77	0.00	471.40	5306	72.37	17.01	72.37	0.00	353.28
5003	166.92	5.21	162.39	0.00	40.76	5134	102.13	6.27	101.24	0.00	117.72	5310	91.87	41.22	88.33	0.00	2028.71
5006	135.92	12.90	135.92	0.00	94.63	5136	93.24	0.00	93.24	0.00	111.43	5312	82.20	39.29	82.20	0.00	90.37
5008	155.42	19.72	152.11	0.00	249.34	5138	87.76	18.90	87.53	0.00	645.75	5321	79.60	6.69	79.60	0.00	92.65
5010	135.72	19.88	133.85	0.00	377.62	5141	96.13	22.29	92.44	2.49	592.25	5324	91.68	41.47	79.75	0.00	1108.16
5012	160.20	22.70	154.34	0.00	232.78	5143	97.61	15.01	93.87	0.00	463.49	5330	96.35	31.96	96.35	0.00	632.45
5015	158.97	15.27	154.38	0.00	300.89	5145	88.13	0.00	88.13	0.00	71.65	5334	89.68	7.45	88.89	0.00	98.94
5017	157.33	18.69	156.18	0.00	189.40	5148	91.54	9.25	865.29	0.00	128.27	5336	87.41	43.82	86.39	0.00	3080.09
5020	146.01	26.95	141.92	0.00	146.23	5150	77.99	9.01	77.42	0.00	160.82	5345	101.45	21.97	99.63	0.00	83.02
5022	133.36	39.00	128.16	0.00	1657.48	5152	80.82	23.49	79.17	0.00	163.00	5351	96.86	44.29	90.10	0.00	221.93

Furthermore, the maximum gap obtained by our *B&C* and the other two methods are 5.31% and 23.99%, respectively. Moreover, considering the 37 instances solved to optimality for the first time in literature by our *B&C*, we can observe that our algorithm determined a new upper bound on 13 of them and proved the optimality of the previously known upper bound on the other 24. For the sake of completeness, we point out that our approach is faster than *BMS21* also on this set of instances with an average running time equal to 77.43 and 1765.25 s, respectively.

Considering the results on the instances with *Dtl* equal to 40 min, we can observe that the state-of-the-art algorithms can optimally solve just 6 instances out of 120 with an average and maximum optimality gap equal to 18.66% and 49.66%, respectively. On the other hand, the performance of the *B&C* seems again to be not significantly affected by the larger *Dtl*. Indeed, it is able to solve 115 instances with an average and maximum optimality gap equal to 0.15% and 10.30%, respectively.

Concerning the 109 instances solved to optimality for the first time in literature by our *B&C*, we can observe that it determined a new upper bound for 87 instances and proved the optimality of the previously known upper bound for the other 22. Finally, concerning the five instances that are still open, the *B&C* improves the best known upper bound for almost all of them (four out of five).

In terms of computation times, a comparison with *BMS21* is possible but not significant since it is not able to solve almost all the instances within the time limit of 1 h. Nevertheless, for the sake of completeness, we highlight that the average computation times for *BMS21* is 3509.08 s. On the other hand, the average computation times of our *B&C* is 535.46 seconds.

Based on these results, it is clear that the proposed *B&C* could be considered the new state-of-the-art algorithm also for the *M20-instances*.

TABLE A5 Results on *POI-instances* with $|C| = 9$.

	RR21			*B&C*				*RR21*			*B&C*		
Id	**UB**	**Time**	**% Gap**	**UB**	**Time**	**%Gap**	**Id**	**UB**	**Time**	**% Gap**	**UB**	**Time**	**% Gap**
poi-10-1-1	173	0.07	0.00	173	0.13	0.00	poi-10-20-3	128	0.1	0.00	128	0.11	0.00
poi-10-1-2	166	0.07	0.00	166	0.07	0.00	poi-10-21-1	132	0.12	0.00	132	0.06	0.00
poi-10-1-3	166	0.1	0.00	166	0.24	0.00	poi-10-21-2	79	0.22	0.00	79	0.31	0.00
poi-10-10-1	175	0.12	0.00	175	0.08	0.00	poi-10-21-3	78	0.24	0.00	78	0.25	0.00
poi-10-10-2	138	0.16	0.00	138	0.28	0.00	poi-10-22-1	153	0.12	0.00	153	0.13	0.00
poi-10-10-3	110	0.3	0.00	110	0.27	0.00	poi-10-22-2	150	0.14	0.00	150	0.39	0.00
poi-10-11-1	169	0.09	0.00	169	0.10	0.00	poi-10-22-3	147	0.15	0.00	147	0.18	0.00
poi-10-11-2	154	0.09	0.00	154	0.08	0.00	poi-10-23-1	182	0.12	0.00	182	0.06	0.00
poi-10-11-3	140	0.15	0.00	140	0.13	0.00	poi-10-23-2	182	0.13	0.00	182	0.07	0.00
poi-10-12-1	182	0.1	0.00	182	0.08	0.00	poi-10-23-3	182	0.15	0.00	182	0.12	0.00
poi-10-12-2	171	0.13	0.00	171	0.28	0.00	poi-10-24-1	168	0.09	0.00	168	0.09	0.00
poi-10-12-3	169	0.12	0.00	169	0.37	0.00	poi-10-24-2	122	0.12	0.00	122	0.23	0.00
poi-10-13-1	170	0.1	0.00	170	0.09	0.00	poi-10-24-3	118	0.12	0.00	118	0.25	0.00
poi-10-13-2	149	0.1	0.00	149	0.22	0.00	poi-10-25-1	169	0.13	0.00	169	0.15	0.00
poi-10-13-3	100	0.19	0.00	100	0.38	0.00	poi-10-25-2	164	0.11	0.00	164	0.34	0.00
poi-10-14-1	162	0.12	0.00	162	0.10	0.00	poi-10-25-3	148	0.1	0.00	148	0.31	0.00
poi-10-14-2	138	0.14	0.00	138	0.22	0.00	poi-10-3-1	178	0.14	0.00	178	0.08	0.00
poi-10-14-3	106	0.17	0.00	106	0.31	0.00	poi-10-3-2	177	0.1	0.00	177	0.14	0.00
poi-10-15-1	195	0.13	0.00	195	0.09	0.00	poi-10-3-3	165	0.08	0.00	165	0.10	0.00
poi-10-15-2	179	0.15	0.00	179	0.16	0.00	poi-10-4-1	178	0.13	0.00	178	0.10	0.00
poi-10-15-3	143	0.14	0.00	143	0.16	0.00	poi-10-4-2	149	0.09	0.00	149	0.29	0.00
poi-10-16-1	174	0.11	0.00	174	0.07	0.00	poi-10-4-3	145	0.29	0.00	145	0.32	0.00
poi-10-16-2	151	0.1	0.00	151	0.07	0.00	poi-10-5-1	150	0.09	0.00	150	0.09	0.00
poi-10-16-3	113	0.12	0.00	113	0.12	0.00	poi-10-5-2	147	0.09	0.00	147	0.14	0.00
poi-10-17-1	201	0.1	0.00	201	0.06	0.00	poi-10-5-3	147	0.14	0.00	147	0.42	0.00
poi-10-17-2	185	0.1	0.00	185	0.09	0.00	poi-10-6-1	178	0.11	0.00	178	0.11	0.00
poi-10-17-3	129	0.13	0.00	129	0.12	0.00	poi-10-6-2	153	0.11	0.00	153	0.13	0.00
poi-10-18-1	166	0.1	0.00	166	0.07	0.00	poi-10-6-3	148	0.21	0.00	148	0.52	0.00
poi-10-18-2	141	0.11	0.00	141	0.10	0.00	poi-10-7-1	158	0.12	0.00	158	0.09	0.00
poi-10-18-3	135	0.24	0.00	135	0.24	0.00	poi-10-7-2	131	0.12	0.00	131	0.42	0.00
poi-10-19-1	174	0.12	0.00	174	0.09	0.00	poi-10-7-3	123	0.26	0.00	123	0.70	0.00
poi-10-19-2	163	0.1	0.00	163	0.10	0.00	poi-10-8-1	133	0.1	0.00	133	0.18	0.00
poi-10-19-3	159	0.1	0.00	159	0.20	0.00	poi-10-8-2	113	0.12	0.00	113	0.18	0.00
poi-10-2-1	175	0.1	0.00	175	0.08	0.00	poi-10-8-3	97	0.37	0.00	97	0.20	0.00
poi-10-2-2	162	0.1	0.00	162	0.15	0.00	poi-10-9-1	189	0.1	0.00	189	0.07	0.00
poi-10-2-3	162	0.14	0.00	162	0.21	0.00	poi-10-9-2	168	0.11	0.00	168	0.10	0.00
poi-10-20-1	155	0.07	0.00	155	0.09	0.00	poi-10-9-3	159	0.17	0.00	165	0.26	0.00
poi-10-20-2	140	0.1	0.00	140	0.16	0.00							

A.3 | POI-instances

The *POI-instances* can be grouped into four different subsets, of 75 instances each, on the basis of the number of customers (9, 19, 29, and 39 customers). Tables A5,A6,A7, and A8 report the detailed results of the comparison of our method with *RR21* on the four different subsets. To be more precise, each table corresponds to a subset of instances with the same number of customers. Then, for each instance, each table shows the upper bound, the running time, and the percentage optimality gap corresponding to each solution method.

Concerning the results on the instance with 9 customers reported in Table A5, we can observe that both methods are able to optimally solve all the instances. Moreover, the two solution methods show similar performance also in terms of running times. Indeed, *RR21* and *B&C* show an average running time of 0.13 and 0.18 s, respectively.

The two methods behave similarly also on the subset of instances with 19 customers, as can be observed from Table A6. Indeed, both methods get the optimal solution on all the instances with an average running time of 3.18 and 6.07 seconds for *RR21* and *B&C*, respectively.

TABLE A6 Results on *POI-instances* with $|C| = 19$.

	RR21			*B&C*				*RR21*			*B&C*		
Id	**UB**	**Time**	**% Gap**	**UB**	**Time**	**%Gap**	**Id**	**UB**	**Time**	**% Gap**	**UB**	**Time**	**% Gap**
poi-20-1-1	226	0.4	0.00	226	1.06	0.00	poi-20-20-3	184	2.02	0.00	184	4.47	0.00
poi-20-1-2	212	0.71	0.00	212	3.39	0.00	poi-20-21-1	205	0.27	0.00	205	1.15	0.00
poi-20-1-3	185	3.67	0.00	185	9.70	0.00	poi-20-21-2	179	0.99	0.00	179	5.04	0.00
poi-20-10-1	213	0.23	0.00	213	1.42	0.00	poi-20-21-3	166	5.93	0.00	166	23.09	0.00
poi-20-10-2	188	2.05	0.00	188	5.61	0.00	poi-20-22-1	235	0.68	0.00	235	1.51	0.00
poi-20-10-3	178	1.27	0.00	178	4.44	0.00	poi-20-22-2	198	7.09	0.00	198	8.11	0.00
poi-20-11-1	205	0.32	0.00	205	1.18	0.00	poi-20-22-3	167	2.91	0.00	167	10.30	0.00
poi-20-11-2	180	1.08	0.00	180	3.20	0.00	poi-20-23-1	227	1.11	0.00	228	4.51	0.00
poi-20-11-3	175	3.17	0.00	175	10.74	0.00	poi-20-23-2	207	3.08	0.00	207	8.23	0.00
poi-20-12-1	225	15.03	0.00	225	2.91	0.00	poi-20-23-3	169	10.15	0.00	169	6.43	0.00
poi-20-12-2	172	7.96	0.00	172	2.86	0.00	poi-20-24-1	234	1.2	0.00	234	3.19	0.00
poi-20-12-3	158	3.46	0.00	158	6.96	0.00	poi-20-24-2	154	0.94	0.00	154	5.24	0.00
poi-20-13-1	208	0.41	0.00	208	0.88	0.00	poi-20-24-3	128	3.89	0.00	128	10.29	0.00
poi-20-13-2	200	3.51	0.00	200	3.88	0.00	poi-20-25-1	230	0.55	0.00	230	2.15	0.00
poi-20-13-3	162	11	0.00	169	5.06	0.00	poi-20-25-2	185	2.43	0.00	185	3.44	0.00
poi-20-14-1	189	1.12	0.00	189	2.96	0.00	poi-20-25-3	170	4.71	0.00	171	10.00	0.00
poi-20-14-2	174	16.54	0.00	174	12.08	0.00	poi-20-3-1	255	0.6	0.00	255	2.46	0.00
poi-20-14-3	159	8.84	0.00	159	23.85	0.00	poi-20-3-2	221	1.41	0.00	221	2.58	0.00
poi-20-15-1	251	1.12	0.00	251	2.80	0.00	poi-20-3-3	198	0.8	0.00	198	7.98	0.00
poi-20-15-2	170	1.67	0.00	170	2.58	0.00	poi-20-4-1	228	0.78	0.00	228	1.24	0.00
poi-20-15-3	157	7.3	0.00	157	6.35	0.00	poi-20-4-2	166	5.24	0.00	166	5.90	0.00
poi-20-16-1	212	0.49	0.00	212	1.48	0.00	poi-20-4-3	146	8.66	0.00	146	6.69	0.00
poi-20-16-2	172	1.1	0.00	172	2.99	0.00	poi-20-5-1	217	2.1	0.00	217	3.09	0.00
poi-20-16-3	171	3.22	0.00	171	4.35	0.00	poi-20-5-2	166	5.91	0.00	166	5.60	0.00
poi-20-17-1	192	2.23	0.00	192	2.79	0.00	poi-20-5-3	153	10.33	0.00	153	11.73	0.00
poi-20-17-2	137	1.34	0.00	137	11.97	0.00	poi-20-6-1	208	0.86	0.00	208	1.49	0.00
poi-20-17-3	134	6.41	0.00	134	15.82	0.00	poi-20-6-2	144	1.19	0.00	144	6.68	0.00
poi-20-18-1	223	0.57	0.00	225	2.60	0.00	poi-20-6-3	130	3.46	0.00	130	11.69	0.00
poi-20-18-2	201	1.05	0.00	201	6.84	0.00	poi-20-7-1	220	0.76	0.00	220	3.54	0.00
poi-20-18-3	178	2.82	0.00	178	22.70	0.00	poi-20-7-2	200	1.07	0.00	200	2.41	0.00
poi-20-19-1	215	1.12	0.00	215	1.04	0.00	poi-20-7-3	200	4.27	0.00	200	8.16	0.00
poi-20-19-2	155	0.92	0.00	155	4.25	0.00	poi-20-8-1	217	1.56	0.00	217	3.66	0.00
poi-20-19-3	146	1.8	0.00	146	7.61	0.00	poi-20-8-2	196	3.19	0.00	196	5.15	0.00
poi-20-2-1	233	0.75	0.00	233	1.83	0.00	poi-20-8-3	178	4.05	0.00	178	14.23	0.00
poi-20-2-2	195	2.27	0.00	196	4.69	0.00	poi-20-9-1	188	0.3	0.00	188	2.19	0.00
poi-20-2-3	172	3.93	0.00	172	6.86	0.00	poi-20-9-2	167	2.79	0.00	167	10.26	0.00
poi-20-20-1	208	0.65	0.00	208	2.33	0.00	poi-20-9-3	158	7.12	0.00	158	13.27	0.00
poi-20-20-2	186	2.25	0.00	186	3.82	0.00							

TABLE A7 Results on *POI-instances* with $|C| = 29$.

	RR21			*B&C*				*RR21*			*B&C*		
Id	**UB**	**Time**	**% Gap**	**UB**	**Time**	**%Gap**	**Id**	**UB**	**Time**	**% Gap**	**UB**	**Time**	**% Gap**
poi-30-1-1	246	21.16	0.00	246	14.66	0.00	poi-30-20-3	168	482.93	0.00	168	3600	2.38
poi-30-1-2	204	110.71	0.00	204	129.05	0.00	poi-30-21-1	233	6.53	0.00	233	11.25	0.00
poi-30-1-3	185	21.82	0.00	185	321.78	0.00	poi-30-21-2	213	90.81	0.00	213	128.80	0.00
poi-30-10-1	248	47.25	0.00	248	36.72	0.00	poi-30-21-3	201	28.63	0.00	201	199.27	0.00
poi-30-10-2	204	74.63	0.00	204	543.56	0.00	poi-30-22-1	255	5.38	0.00	255	13.30	0.00
poi-30-10-3	189	175.5	0.00	189	379.35	0.00	poi-30-22-2	217	101.55	0.00	217	43.38	0.00
poi-30-11-1	249	60.31	0.00	249	38.03	0.00	poi-30-22-3	209	160.06	0.00	209	107.71	0.00
poi-30-11-2	200	73.34	0.00	200	106.18	0.00	poi-30-23-1	232	181.92	0.00	232	99.48	0.00
poi-30-11-3	189	139.91	0.00	189	124.91	0.00	poi-30-23-2	197	709.05	0.00	197	133.59	0.00
poi-30-12-1	258	1.88	0.00	258	19.63	0.00	poi-30-23-3	180	39.12	0.00	180	264.68	0.00
poi-30-12-2	190	16.21	0.00	190	66.34	0.00	poi-30-24-1	219	21.66	0.00	219	20.64	0.00
poi-30-12-3	180	73.77	0.00	180	527.78	0.00	poi-30-24-2	185	37.68	0.00	185	345.63	0.00
poi-30-13-1	256	21.42	0.00	256	10.76	0.00	poi-30-24-3	177	63.39	0.00	177	443.47	0.00
poi-30-13-2	205	52.99	0.00	205	61.21	0.00	poi-30-25-1	259	2.39	0.00	259	23.12	0.00
poi-30-13-3	199	181.04	0.00	199	386.96	0.00	poi-30-25-2	234	46.16	0.00	234	84.33	0.00
poi-30-14-1	229	7.17	0.00	229	8.06	0.00	poi-30-25-3	227	197.12	0.00	227	290.78	0.00
poi-30-14-2	186	117.55	0.00	186	55.74	0.00	poi-30-3-1	224	75.84	0.00	224	31.92	0.00
poi-30-14-3	177	200.46	0.00	177	246.43	0.00	poi-30-3-2	192	39.39	0.00	192	42.93	0.00
poi-30-15-1	214	57.29	0.00	214	26.37	0.00	poi-30-3-3	192	245.5	0.00	192	55.33	0.00
poi-30-15-2	189	73.75	0.00	189	112.67	0.00	poi-30-4-1	260	6.67	0.00	260	34.37	0.00
poi-30-15-3	184	44.88	0.00	184	141.59	0.00	poi-30-4-2	206	25.13	0.00	206	378.35	0.00
poi-30-16-1	246	4.99	0.00	246	11.58	0.00	poi-30-4-3	158	25.69	0.00	158	156.68	0.00
poi-30-16-2	202	50.72	0.00	202	74.26	0.00	poi-30-5-1	230	9.56	0.00	230	14.32	0.00
poi-30-16-3	196	83.64	0.00	196	101.33	0.00	poi-30-5-2	193	54.89	0.00	193	311.91	0.00
poi-30-17-1	255	33.82	0.00	255	22.71	0.00	poi-30-5-3	188	80.25	0.00	188	217.22	0.00
poi-30-17-2	192	26.28	0.00	192	47.16	0.00	poi-30-6-1	251	101.66	0.00	251	35.21	0.00
poi-30-17-3	174	32.11	0.00	174	102.18	0.00	poi-30-6-2	208	76.1	0.00	208	256.35	0.00
poi-30-18-1	257	28.8	0.00	257	32.05	0.00	poi-30-6-3	194	125.45	0.00	194	52.24	0.00
poi-30-18-2	219	159.4	0.00	219	46.71	0.00	poi-30-7-1	211	12.19	0.00	211	18.97	0.00
poi-30-18-3	179	53.9	0.00	179	954.68	0.00	poi-30-7-2	190	14.73	0.00	190	116.52	0.00
poi-30-19-1	232	6.59	0.00	232	32.95	0.00	poi-30-7-3	182	72.03	0.00	182	148.21	0.00
poi-30-19-2	203	9.2	0.00	203	149.00	0.00	poi-30-8-1	239	5.93	0.00	239	11.02	0.00
poi-30-19-3	202	53.92	0.00	202	63.95	0.00	poi-30-8-2	200	14.55	0.00	200	111.55	0.00
poi-30-2-1	263	55.31	0.00	263	27.75	0.00	poi-30-8-3	195	109.96	0.00	195	256.07	0.00
poi-30-2-2	200	78.26	0.00	200	105.56	0.00	poi-30-9-1	259	66.9	0.00	259	25.39	0.00
poi-30-2-3	180	120.08	0.00	180	214.48	0.00	poi-30-9-2	210	154.03	0.00	210	185.48	0.00
poi-30-20-1	228	16.14	0.00	228	39.31	0.00	poi-30-9-3	193	886.05	0.00	193	498.22	0.00
poi-30-20-2	178	158.56	0.00	178	768.54	0.00							

Concerning the instances instead with 29 customers, we can observe from Table A7 that *RR21* is still able to solve to optimality all the instances. At the same time, our *B&C* is able to solve 74 out of 75 instances. However, the maximum optimality gap is equal to 2.38%. In terms of running time, we can observe that the average running time is equal to 92.29 s for *RR21* and 198.94 s for *B&C* (including the unsolved instance within the time limit).

Finally, we can observe from Table A8 that none of the two algorithms is able to optimally solve all the 75 instances with 39 customers within the time limit. In particular, *RR21* is able to solve 61 instances while our *B&C* determines the optimal solution on 57 instances. However, despite solving fewer instances to optimality, the *B&C* shows a lower average percentage gap (0.58%) than *RR21* (5.48%). Moreover, the *B&C* performs better than *RR21* also in terms of the largest percentage gap (6.22% vs. 51.01%). Instead, the running times are similar, with an average running time of 1475.88 and 1482.69 s for *RR21* and *B&C*, respectively.

TABLE A8 Results on *POI-instances* with $|C| = 39$.

	RR21			*B&C*				*RR21*			*B&C*		
Id	**UB**	**Time**	**% Gap**	**UB**	**Time**	**%Gap**	**Id**	**UB**	**Time**	**% Gap**	**UB**	**Time**	**% Gap**
poi-40-1-1	283	1364.79	0.00	283	115.45	0.00	poi-40-20-3	216	2545.92	0.00	216	719.56	0.00
poi-40-1-2	213	563.64	0.00	214	3600	0.93	poi-40-21-1	269	2148.13	0.00	269	117.65	0.00
poi-40-1-3	194	3267.08	0.00	194	1002.15	0.00	poi-40-21-2	227	309.95	0.00	227	3077.10	0.00
poi-40-10-1	259	403.45	0.00	259	216.47	0.00	poi-40-21-3	218	802.95	0.00	218	3600	1.38
poi-40-10-2	220	74.69	0.00	220	402.12	0.00	poi-40-22-1	286	50.98	0.00	286	163.69	0.00
poi-40-10-3	218	696.9	0.00	218	971.05	0.00	poi-40-22-2	243	413.37	0.00	243	1017.52	0.00
poi-40-11-1	266	3600	0.75	266	351.62	0.00	poi-40-22-3	339	3600	34.22	224	1302.18	0.00
poi-40-11-2	222	527.99	0.00	222	513.92	0.00	poi-40-23-1	395	3600	26.84	302	609.36	0.00
poi-40-11-3	210	378.64	0.00	210	328.44	0.00	poi-40-23-2	395	3600	38.48	244	480.90	0.00
poi-40-12-1	262	35.93	0.00	262	47.25	0.00	poi-40-23-3	230	1297.49	0.00	230	546.49	0.00
poi-40-12-2	222	1405.59	0.00	222	1164.56	0.00	poi-40-24-1	244	3227.38	0.00	244	1726.32	0.00
poi-40-12-3	209	2534	0.00	209	1162.17	0.00	poi-40-24-2	215	125.84	0.00	215	391.89	0.00
poi-40-13-1	266	988.14	0.00	266	1408.90	0.00	poi-40-24-3	212	1537.43	0.00	212	1076.67	0.00
poi-40-13-2	222	326.15	0.00	222	363.11	0.00	poi-40-25-1	417	3600	33.57	281	1478.26	0.00
poi-40-13-3	363	3600	42.15	210	3600	0.95	poi-40-25-2	210	361.2	0.00	210	667.90	0.00
poi-40-14-1	287	64.94	0.00	287	104.26	0.00	poi-40-25-3	190	2450.66	0.00	190	3600	1.05
poi-40-14-2	229	673.8	0.00	229	1686.05	0.00	poi-40-3-1	282	438.14	0.00	282	51.12	0.00
poi-40-14-3	210	254.43	0.00	210	1063.50	0.00	poi-40-3-2	416	3600	43.51	235	1412.65	0.00
poi-40-15-1	279	215.33	0.00	279	3600	1.43	poi-40-3-3	416	3600	48.08	216	3600	3.24
poi-40-15-2	234	302.24	0.00	234	550.87	0.00	poi-40-4-1	284	3600	0.35	288	1132.86	0.00
poi-40-15-3	346	3600	33.53	231	3600	4.33	poi-40-4-2	234	933.35	0.00	234	3600	1.71
poi-40-16-1	270	1918.15	0.00	270	3600	1.85	poi-40-4-3	210	565.64	0.00	210	1602.15	0.00
poi-40-16-2	210	952.09	0.00	210	3600	0.95	poi-40-5-1	285	810.61	0.00	285	508.73	0.00
poi-40-16-3	201	229.68	0.00	201	3202.26	0.00	poi-40-5-2	231	2302.31	0.00	231	1401.09	0.00
poi-40-17-1	285	187.93	0.00	285	752.74	0.00	poi-40-5-3	203	450.09	0.00	203	3600	3.41
poi-40-17-2	246	79.75	0.00	246	707.20	0.00	poi-40-6-1	240	2325.71	0.00	240	921.17	0.00
poi-40-17-3	359	3600	32.59	242	1142.52	0.00	poi-40-6-2	192	1270.8	0.00	192	3600	3.65
poi-40-18-1	391	3600	25.32	297	3600	1.68	poi-40-6-3	181	1351.85	0.00	181	3600	1.10
poi-40-18-2	239	487.04	0.00	241	3600	6.22	poi-40-7-1	271	3600	0.74	271	593.41	0.00
poi-40-18-3	221	68.9	0.00	221	3600	5.43	poi-40-7-2	206	506.34	0.00	206	2602.22	0.00
poi-40-19-1	236	560.59	0.00	236	715.17	0.00	poi-40-7-3	396	3600	51.01	194	3600	1.55
poi-40-19-2	204	2748.63	0.00	204	400.35	0.00	poi-40-8-1	277	494.89	0.00	277	594.32	0.00
poi-40-19-3	204	2452.33	0.00	204	260.69	0.00	poi-40-8-2	228	628.96	0.00	228	622.08	0.00
poi-40-2-1	269	60.64	0.00	269	175.37	0.00	poi-40-8-3	220	972.64	0.00	220	667.60	0.00
poi-40-2-2	218	1092.7	0.00	218	1128.46	0.00	poi-40-9-1	289	422.85	0.00	289	202.98	0.00
poi-40-2-3	204	834.17	0.00	204	2146.83	0.00	poi-40-9-2	226	924.52	0.00	226	924.76	0.00
poi-40-20-1	262	2285.6	0.00	262	363.29	0.00	poi-40-9-3	208	1252.64	0.00	208	503.69	0.00
poi-40-20-2	220	1332.06	0.00	220	398.77	0.00							