The Green Vehicle Routing Problem with Capacitated Alternative Fuel Stations

M. Bruglieri[a,*], S. Mancini[b], O. Pisacane[c]

[a] *Dipartimento di Design, Politecnico di Milano*
[b] *Dipartimento di Matematica e Informatica, Universitá di Cagliari*
[c] *Dipartimento di Ingegneria dell'Informazione, Universitá Politecnica delle Marche*

Abstract

In this paper, we introduce the Green Vehicle Routing Problem with capacitated Alternative Fuel Stations (AFSs), a more realistic variant of the Green Vehicle Routing Problem where the capacity of the AFSs is addressed. Two Mixed Integer Linear Programming formulations, one based on arc-variables and one on path-variables are presented. In order to reduce the computational time required to solve the problem, two variants of an exact cutting planes method are proposed. Computational experiments have been carried out on both some benchmark instances and challenging realistic instances for which the capacity of the AFSs is a crucial issue.

Keywords: Vehicle Routing Problem, Alternative Fuel Vehicles, Mixed Integer Linear Programming, Cutting Planes

1. Introduction

Nowadays, finding solutions to reduce environmental pollution is becoming a crucial issue. Recent statistical studies have remarked that from 2010 to 2017 the rate of atmospheric CO_2 increased of about 5% (www.co2.earth). In particular, the transport sector produces about 23% of the global CO_2 and this rate is expected to double in 2050. It is not surprising that recent rules imposed by the European Commission tend to incentivize the use of *Alternative Fuel Vehicles* (AFVs), i.e., vehicles that use alternative fuel (e.g.,

*Corresponding author: maurizio.bruglieri@polimi.it

methane, electricity), instead of traditional Internal Combustion Engine Vehicles (ICEVs).

Besides a significant reduction of both the harmful emissions and the fossil fuels dependency, the use of the AFVs allows reaching Limited Traffic Zones, providing, among other things, more efficient door-to-door distribution services. Although the purchase cost of AFVs is still higher compared to that of ICEVs, its operating cost is by far lower. For example, a study carried out by [12] shows that the operating cost of a conventional diesel truck (e.g., Isuzu N-Series) is about \$0.23/miles against \$0.09/miles of an electric propulsion one (e.g., Navistar E-star).

However, the very limited driving range still represents the Achilles' heel for the AFVs. In fact, they require several stops at *Alternative Fuel Stations* (AFSs) along their route. Moreover, since AFSs are not widespread across the territory, refueling stops should be a priori planned to prevent drivers to remain stuck along their routes.

An emerging topic which is attracting the attention of many academics is the *Green Vehicle Routing Problem* (GVRP). The GVRP, introduced in the literature by the seminal work of [10], refers to the problem of routing a fleet of AFVs, based at a common depot, to serve a set of customers, minimizing the total travel distance. During the trips, the AFVs can be refueled, even more than once, at AFSs and each time, to their full capacity.

Each AFV, fully refueled, leaves the depot and returns to it, within a maximum duration that can depend on many factors, among which, for instance, the working conditions of drivers. For each customer, a *service time* is a-priori known. Similarly, a *refueling time* to take into account stops at each AFS is specified. No cargo capacity nor service time windows at the customers are considered.

Since fuel consumption is assumed to be linearly proportional to the travel distance, the maximum distance an AFV can travel without refueling can be easily derived from its tank capacity.

In the traditional GVRP, it is implicitly assumed that the number of AFVs can be simultaneously refueled at the same AFS is unlimited. Firstly, such an assumption is unrealistic since, in real life, each AFS owns only a limited number of refueling pumps. Therefore, we introduce a variant of the GVRP where the number of refueling pumps available in each AFS is considered. Under this new assumption, queues at AFSs may occur and must be considered in the model. Indeed, a possible queuing at an AFS may increase the actual refueling time, yielding to infeasible routes. This

could happen especially when the refueling times are high (e.g., in the case of electric vehicles) and/or when the maximum route duration is very tight.

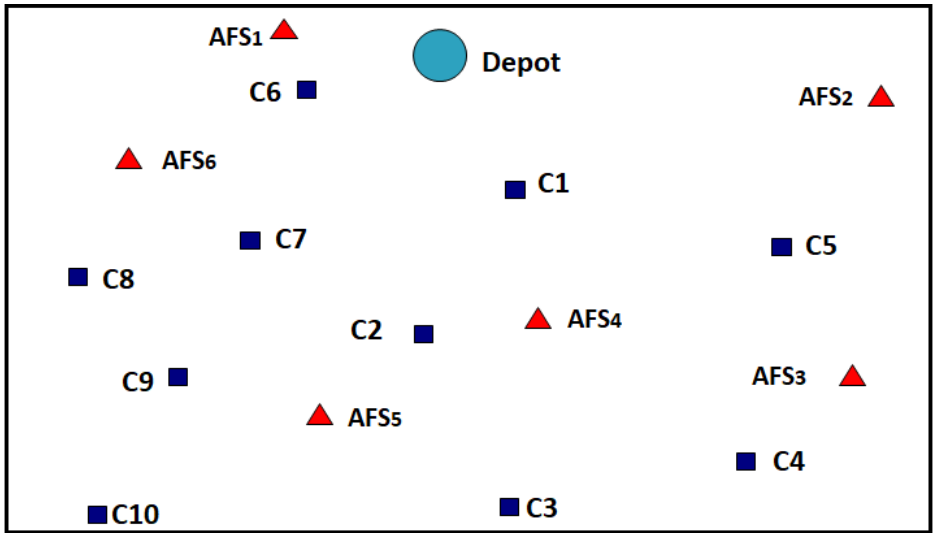

Figure 1: Scenario in which the AFS capacity is not a crucial issue

Figure 1 refers to a situation, frequently arising in urban contexts, where AFSs are perfectly integrated in the customers area. In such a case, a limited number of fueling pumps per station does not represent a crucial issue. On the contrary, Figure 2 shows a realistic scenario related to an area in the northern Italy concerning the horizontal stretch of highway among the cities of Asti, Alessandria and Tortona. In such a scenario, the AFS capacity may become a crucial issue because the AFVs require to refuel before reaching the customers area. Therefore, overlaps among AFVs at the same AFS may frequently occur.

In this work, we introduce a variant of the traditional GVRP, i.e., the *GVRP with Capacitated AFSs* (GVRP-CAFS) where only a limited number, η_s, of refueling pumps are available at an AFS s. Therefore at most η_s AFVs can simultaneously refuel in s. The GVRP-CAFS fits well in two main kinds of scenario. The first one is that where all AFSs belong to the company that wants to route the AFVs. The second scenario is the one where the AFSs are public but the companies can reserve in advance the use of the refueling pumps of each AFS, in order to avoid unpredictable waiting times at the AFSs due to the fact that all pumps are busy. Of course, also a hybrid

Figure 2: Scenario in which the AFS capacity plays a crucial role

scenario where the companies can use both public AFSs and their private ones is possible.

The main contributions of this paper are:

- the introduction of a more realistic variant of the GVRP
- an Arc based Mixed Integer Linear Programming model;
- a more efficient Path based Mixed Integer Programming formulation;
- the modeling of (also multiple) time windows associated with refueling pumps;
- two variants of an exact cutting planes approach for the Path based model;
- the generation of a benchmark set of challenging instances for the GVRP-CAFS.

The rest of the paper is organized as follows. Section 2 describes the state-of-the-art contributions on the GVRP and its variants while, Section 3 states the GVRP-CAFS. Section 4 introduces an Arc based mathematical formulation of the GVRP-CAFS while Section 5 proposes a Path based model and its linearization. Section 6 describes the two variants of the exact cutting planes method for solving the GVRP-CAFS in a reasonable amount of time. In section 7, a set of benchmark challenging instances is introduced and numerical results are discussed. Finally, Section 8 draws some conclusions and outlines future research directions.

2. Literature review

The problem of efficiently routing a fleet of AFVs is currently under investigation and many contributions already exist in the literature, as proven by several surveys on the topic (e.g., [21], [8] and [3]).

The GVRP belongs to the general class of Vehicle Routing Problems (VRPs) and it was introduced in the literature by the work of [10] in which the authors mathematically model it via Mixed Integer Linear Programming (MILP). They also design two construction heuristics: a Modified Clarke and Wright Savings algorithm and a Density-Based Clustering algorithm. Numerical comparisons are also proposed and discussed on a set of benchmark instances ad hoc generated for the GVRP. In the same year, [16] proposed a bi-objective GVRP, including advancements on the way of estimating the CO_2 emissions. For solving the benchmark instances, they apply the NSGA-II evolutionary algorithm.

In [11], the GVRP is addressed under the hypothesis that the fleet is made up by only electric vehicles. Therefore, they consider the possibility of using different recharging technologies, each associated with a specific cost. Both constructive and improving heuristics are designed and embedded in a Simulated Annealing framework. Both user and operator costs, together with the environment state, are introduced in the GVRP by [17] and a neuro-fuzzy approach is proposed for solving this problem. In [27], the Electric Vehicle Routing Problem with Time Windows (EVRPTW), i.e., the problem of routing a fleet of electric vehicles, is mathematically formulated, considering that each customer also defines a time window for being served, each vehicle has a limited cargo capacity and the need of being recharged at the stations during the trips. For EVRPTW, the authors also design a meta-heuristic that combines a Variable Neighborhood Search together with a Tabu Search. A set of benchmark instances are also ad hoc generated for the new problem. EVRPTW is addressed as a VRP with intermediate stops and an Adaptive Neighborhood Search is proposed in [28]. Instead, in [7], a new variant of the EVRPTW, introducing the possibility of partially recharging the vehicles at the station, is addressed. Therefore, a time-effective MILP model is proposed and large-sized instances are also solved via a matheuristic, i.e., a Variable Neighborhood Search combined with the Local Branching method. In [15], the authors propose a modeling framework for electric vehicle service systems, proving that dynamic routing policies can maximize the throughput and guarantee a system stability by incorporating also information on

the vehicle arrival and service as well as on the routing processes. Recently, two new variants of the EVRPTW have been proposed in the literature. One variant concerns the problem of sizing a fleet of mix (or heterogeneous) electric vehicles together with their routing, i.e., the Electric Fleet Size and Mix Routing Problem with Time Windows and Recharging Stations [14]. For such a new variant, the authors propose both a branch-and-price algorithm and a hybrid heuristic that combines an Adaptive Large Neighborhood Search with an embedded local search and a labeling procedure for intensification. Another variant is related to the routing of a mix fleet made up by conventional, plug-in hybrid and electric vehicles [13]. The authors propose a metaheuristic that integrates the genetic algorithm framework with a local and large neighborhood search.

A more efficient MILP formulation is proposed in the work of [4] in which the AFSs are only implicitly considered. In this way, the authors prove that the total number of variables and constraints are significantly reduced with regard to the traditional formulation of the problem. In addition, the number of variables is further reduced by pre-computing, for each pair of customers, an efficient subset of AFSs, i.e., the ones may be actually used in an optimal solution. Also, in [18], a new mathematical formulation of the GVRP is proposed, by adapting the Miller Tucker Zemlin (MTZ) capacity and subtour elimination constraints already proposed for the Traveling Salesman Problem. An exact solution approach, based on a branch-and-bound algorithm, is designed and a Simulated Annealing approach is used for finding upper bounds.

A two phase heuristic approach is proposed in [25]. The authors firstly build a set of feasible routes through the combination of a randomized route-first cluster-second heuristic with an optimal AFSs insertion procedure. Then, such a route is given in input to a set partitioning model for assembling a GVRP solution. Very recently, in [1], a Variable Neighborhood Search heuristic is designed for the GVRP showing that it is competitive with the other already existing heuristics.

Four variants of the traditional EVRPTW are introduced and addressed in [9]. In particular, assuming that only full battery recharges are allowed, in the first variant, at most a single recharge per route is permitted while, this constraint is relaxed in the second variant. Moreover, the two variants are studied under the hypothesis that partial recharges are also permitted. For each variant, the authors propose an exact branch-and-price-and-cut algorithm.

In [22], a GVRP with simultaneous pickups and deliveries and with also time windows is addressed, introducing a comprehensive modal emission model for taking into account both the fuel consumption and the emissions. An extension of the traditional GVRP is introduced in [23] in which the fleet is made up by hybrid vehicles, i.e., vehicles that can switch from the electric mode to the traditional fuel at any time during their trips. The author mathematically formulates the problem of routing such a fleet for minimizing the total travel distance and also penalizing the use of the traditional fuel. For this problem, the author also designs a Large Neighborhood Search based matheuristic. A variant of the EVRP is proposed in [26] considering nonlinear charging functions.

A new mathematical formulation of the GVRP is described in [20], based on the Reformulation-Linearization Technique. In order to fix some binary variables of the formulation, the authors also describe some pre-processing conditions. Computational results are compared with the ones obtained by both [10] and [18] on the only small/medium-sized instances. While, in [6], the problem introduced in [7] is addressed via a three-phase matheuristic in which a feasible solution is firstly found by solving the MILP modeling a GVRP with capacity and time windows constraints. Secondly, such a solution becomes the input of the EVRPTW with partial recharges, by fixing some routing variables coherently. In the third phase, the EVRPTW with partial recharges, starting from that feasible solution, is solved via a Variable Neighborhood Search with a Local Branching method.

Very recently, in [2], a new exact algorithm for solving the GVRP is proposed. In particular, the authors model the G-VRP as a set partitioning problem in which the columns are feasible routes, i.e., simple circuits on a properly built multigraph. The proposed approach is suitable to solve to optimality also instances with up to 110 customers. The concept of paths in the G-VRP has been introduced in [5] where a path-based formulation and a matheuristic based on it have been proposed.

Although all the above cited works clearly show a real interest in the GVRP, to the best of our knowledge, no contributions have been already proposed for including the capacity of the AFS such as the limitation on the number of AFVs that can be simultaneously refueled at the same station.

Table 1: Nomenclature of the GVRP-CAFS

Set	**Meaning**
I	set of customers
F	set of AFSs
$N = I \cup F \cup \{0\}$	set of nodes
$A = (i, j), \forall i, j \in N$	set of arcs
Parameter	**Meaning**
0	depot
m	number of available AFVs
v	average AFV speed
Q	maximum fuel capacity for each AFV
r	fuel consumption rate
D_{max}	distance an AFV can travel without refueling
T_{max}	maximum route duration
t_{ij}	travel time to go from node i to node j
d_{ij}	travel distance between node i and node j
p_s	refueling time at AFS $s \in F$
p_i	service time at customer $i \in I$
η_s	number of fueling pumps at AFS $s \in F$

3. Notation and assumptions

Similarly to the GVRP, the GVRP-CAFS is formally represented on a complete directed graph $G = (N, A)$, where the set N of nodes contains the set I of customers to be served, the set F of AFSs and the depot, denoted by 0. For each arc $(i, j) \in A$, a travel time t_{ij} as well as a distance d_{ij} are known. Each AFV can perform a route starting from the depot and ending at it, without exceeding the maximum duration T_{max}. The time p_s spent at each AFS s to fully refuel an AFV to its maximum capacity Q is considered, independently of the amount of fuel needed. If the depot is also an AFS, as in the instances of [10], it is indicated by s_0. Moreover, it is assumed that each AFV leaves the depot fully refueled and the time spent for the initial refuel, p^{start}, has to be subtracted from T_{max}. Similarly, for each customer $i \in I$, the service time p_i is given. A fictitious service time $p_0 = 0$ is assigned also to the depot, when it is not considered as an AFS. For each AFV, the average speed v is also given. Since the fuel consumption is assumed to be linearly proportional to the travel distance through a rate r, the maximum

distance D_{max} an AFV can travel without refueling is computed as Q/r. Finally, for each AFS $s \in F$, a limited number η_s of fueling pumps is given, i.e., a limitation on the number of AFVs can be simultaneously refueled at s.

Table 1 summarizes all the sets and parameters introduced for describing the GVRP-CAFS.

4. An Arc Based Formulation for the GVRP-CAFS

We model the capacity constraint on the AFSs introducing a node for each fueling pump at each AFS. Therefore, we introduce η_s fueling pump nodes $\forall s \in F$. In order to allow that the same fueling pump can be used q times, we introduce q clones (i.e., dummy copies) of the node representing it. If only one refuel per route is necessary, an upper bound on q is given by $\lceil \frac{m}{\eta_s} \rceil$. More generally, if μ is the maximum number of refuelings necessary per route, an upper bound on q is given by $\lceil \frac{\mu m}{\eta_s} \rceil$.

The additional notation used in formulating the GVRP-CAFS is defined as follows.

- Φ_s: ordered set of fueling pumps at AFS s
- $\Pi = \cup_{s \in F} \Phi_s$
- $\tilde{F}_h$: ordered set of clones of fueling pumps h, $\forall h \in \Pi$
- $\tilde{F}_h^{last} = \tilde{F}_h \setminus \{last(\tilde{F}_h)\}$, $\forall h \in \Pi$ (where operator $last(\ )$ selects the last element of the set considered).
- $\hat{F} = \cup_{h \in \Pi} \tilde{F}_h$
- $\hat{N} = I \cup \hat{F} \cup \{0\}$
- $\Omega = \{i \in \hat{N}, j \in \hat{N} : i \neq j,\ d_{0i} + d_{ij} + d_{j0} \leq T_{max}\, v,\ d_{ij} \leq D_{max}$ and $\nexists s \in F : i \in \tilde{F}_{h_1}, j \in \tilde{F}_{h_2},\ h_1, h_2 \in \Phi_s\}$ (the last condition corresponds requiring that i and j are not clones of fueling pumps of the same station).

The variables of the problem are now described.

- x_{ij}: binary variable equal to 1 if a vehicle travels from node i to node j and 0 otherwise, $\forall (i,j) \in \Omega$

- y_i: fuel level at node i, $\forall i \in \hat{N}$ (if $i \in \hat{F} \cup \{0\}$, it is the fuel level after refueling);
- τ_i: arrival time at node i, $\forall i \in I \cup \hat{F}$ and starting time from the depot for $i = 0$

The mathematical programming formulation of the GVRP-CAFS for the scenario with private AFSs (first scenario of Section 1) is the following:

$$\min \sum_{(i,j)\in\Omega} d_{ij}x_{ij} \tag{1}$$

s.t.

$$\sum_{(i,j)\in\Omega} x_{ij} = 1 \quad \forall\, i \in I \tag{2}$$

$$\sum_{(i,j)\in\Omega} x_{ij} \leq 1 \quad \forall\, i \in \hat{F} \tag{3}$$

$$\sum_{(j,i)\in\Omega} x_{ji} = \sum_{(i,j)\in\Omega} x_{ij} \quad \forall\, j \in \hat{N} \tag{4}$$

$$\sum_{(0,j)\in\Omega} x_{0j} \leq m \tag{5}$$

$$\tau_j \geq \tau_i + (p_i + t_{ij})x_{ij} - T_{max}(1 - x_{ij}) \quad \forall (i,j) \in \Omega : j \neq 0 \tag{6}$$

$$\tau_0 \geq p^{start} \tag{7}$$

$$\tau_j \leq T_{max} - (t_{j0} + p_j) \quad \forall j \in N : (j,0) \in \Omega \tag{8}$$

$$y_j \leq y_i - rd_{ij}x_{ij} + Q(1 - x_{ij}) \quad \forall (i,j) \in \Omega : j \in I \tag{9}$$

$$y_j = Q \quad \forall j \in \hat{F} \cup \{0\} \tag{10}$$

$$y_i \geq \sum_{(i,j)\in\Omega : j\in\hat{F}\cup\{0\}} rd_{ij}x_{ij} \quad \forall i \in \hat{N} \tag{11}$$

$$\tau_{\tilde{h}+1} \geq \tau_{\tilde{h}} + p_s - (T_{max} + p_s)(1 - \sum_{i \in \hat{N}:(i,\tilde{h}+1)\in\Omega} x_{i\tilde{h}+1}) \ \ \forall s \in F, \forall \tilde{h} \in \cup_{h\in\Phi_s} \tilde{F}_h^{last} \tag{12}$$

$$\sum_{i \in \hat{N}:(i,\tilde{h}_1)\in\Omega} x_{i\tilde{h}_1} \geq \sum_{i \in \hat{N}:(i,\tilde{h}_2)\in\Omega} x_{i\tilde{h}_2} \ \ \forall h \in \Pi, \forall \tilde{h}_1, \tilde{h}_2 \in \tilde{F}_h : \tilde{h}_1 < \tilde{h}_2 \tag{13}$$

$$\sum_{i \in \hat{N}, s_1 \in \tilde{F}_{h_1}:(i,h_1)\in\Omega} x_{ih_1} \geq \sum_{i \in \hat{N}, s_2 \in \tilde{F}_{h_2}:(i,h_2)\in\Omega} x_{ih_2} \ \ \forall s \in F, \forall h_1, h_2 \in \Phi_s : h_1 < h_2 \tag{14}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in \Omega \tag{15}$$

$$y_i \geq 0 \quad \forall i \in \hat{N} \tag{16}$$

$$\tau_i \geq 0 \quad \forall i \in \hat{N} \tag{17}$$

The objective function is defined in (1) and corresponds to the minimization of the total travel distance. Constraints (2) ensure that each customer is visited exactly once, while constraints (3) imply that each cloned fueling pump can be visited at most once. The flow conservation constraints (4) imply that every node $j \in \hat{N}$, i.e., either a customer or an AFS or the depot, entered by an AFV has to be left by the same AFV. Constraint (5) imposes that the maximum number of available AFVs is not exceeded. The arrival time at each node is ruled by constraints (6) that also prevent subtours. Constraint (7) imposes that the starting time from the depot must be not lower than the time spent for the initial refueling. Constraints (8) ensure that each vehicle returns to the depot without exceeding the maximum route duration T_{max}. The tank level, upon arrival at each customer, is ruled by constraints (9). Constraints (10) set the value of the tank level equal to Q at the departure from the depot and after a visit to a cloned fueling pump. Constraints (11) guarantee that if either a clone j of a fueling pump or the depot are visited after a customer i, the tank level at i must be enough to reach j. Constraints (12) model the capacity constraints ensuring that for each clone s of a fueling pump, if its next clone, $s+1$, is used, the visiting

time of the latter is postponed at least by the refueling time. Of course, such constraints are not imposed for the last clone of each fueling pump. On the other hand, constraints (13) ensure that the clones are used in an increasing order to avoid equivalent solutions. Similarly, to avoid equivalent solutions in using the fueling pumps, constraints (14) ensure that also the fueling pumps are used in an increasing order. Finally, constraints (15)-(17) provide the variables nature.

This model can be easily extended to the scenario with public AFSs (second scenario of Section 1) where the fueling pumps reservation has to be addressed. In this case, to take into account that AFSs are also employed by other users, a time window $[e_h, l_h]$ can be introduced for each pump $h \in \Pi$ where e_h and l_h are the earliest and the latest time allowed for refueling, respectively. Moreover, the following constraints have to be added to the previous formulation:

$$e_h \leq \tau_i \leq l_h \quad \forall i \in \tilde{F}_h, \forall h \in \Pi \tag{18}$$

In addition, multiple time windows associated with each pump h can be easily addressed by our model introducing, for each time window, a number of clones of h equal to the minimum between $\lceil \frac{\mu m}{\lambda+1} \rceil$ and the floor of the ratio between the time window width and the refueling time. In particular, μ is the maximum number of refuelings necessary per route and λ is the number of the other pumps of the same AFS, which h belongs to, having time window including that of h. In this way, the problem is reduced to the case with a single time window.

5. A Path Based Formulation for the GVRP-CAFS

In this section, we propose a Path based Mixed Integer Programming (MIP) formulation for the GVRP-CAFS. In fact, each feasible route can be seen as the combination of *paths*, each one handling a subset of customers without intermediate stops at AFSs. Each path can link the depot or an AFS with the depot or another AFS. In particular, the path linking the depot with itself is a route without intermediate stops. Moreover, for each path k, the origin (starting node) s_k, the destination (arrival node) a_k, the travel distance d_k and the elapsed time γ_k are known. In particular, γ_k is computed as the sum of the total travel time of path k and the service times at nodes visited by it. For example, with reference to the scenario in Figure 1, two paths are

$(Depot, C1, C5, AFS_4)$ and $(AFS_4, C2, C7, Depot)$. Therefore, a route is the composition of these two paths: $(Depot, C1, C5, AFS_4, C2, C7, Depot)$.

Algorithm 1 Generation of the feasible non-dominated paths

Input:
Set of customers I, set of AFSs F, depot 0
Number of vehicles m
Maximum path distance D_{max}
Maximum route duration T_{max}.
Output: set of feasible non-dominated paths K

Let $K := \emptyset$;
for $(i, j) \in \{N \setminus I\} \times \{N \setminus I\}$ **do**
  Let $k := (i, j)$;
  if $feasible(k)$ **then**
    if $i = 0$ and $j = 0$ **then**
      $K := K \cup \{k\}$;
    else
      $K' := replicate(k, m)$;
      $K := K \cup K'$;
    end if
    Let $K' := \{(i, j)\}$;
    while $K' \neq \emptyset$ **do**
      $K' := extend(K', D_{max}, T_{max})$
      if $K' \neq \emptyset$ **then**
        $K := K \cup K'$;
        $K := removeDominated(K, K')$
      end if
    end while
  end if
end for

The set K contains all feasible non-dominated paths. A path k is feasible (*feasibility rules*) if: $d_k \leq D_{max}$ and $\gamma_k + p_{s_k} + p_{a_k} + t_{0s_k} + t_{a_k 0} + p^{start} \leq T_{max}$. Moreover, a feasible path k_1 dominates a feasible path k_2 (*dominance rules*) if: $s_{k_1} = s_{k_2}$, $a_{k_1} = a_{k_2}$, they handle the same customers and $d_{k_2} \geq d_{k_1}$.

Algorithm 1 outlines the main steps of the procedure that generates all feasible non-dominated paths. In particular, the $feasible$ routine returns $TRUE$ if the path k respects the *feasibility rules* given above; $FALSE$, otherwise. The *replicate* routine returns as many copies of the feasible path

k as the number of vehicles (m). While, the routine *extend* receives a set of feasible paths K', from origin i to destination j. It returns a new set of feasible paths from i to j, each obtained from a path in K' by adding a new customer, as the last served. For the sake of clarity, the routine initially receives only the path (i, j). Then, it returns all feasible paths of type (i, c, j) $\forall c \in I$. At the second iteration, the routine receives the set of feasible paths of type (i, c, j) and for each of them, it generates all feasible paths of type (i, c, c_1, j) $\forall c_1 \in I$. It ends when no feasible paths of a certain type can be found. Finally, the routine *removeDominated* removes, from the current set of feasible non-dominated paths K, all ones that are dominated after adding the new paths contained in K'.

Each path k belonging to K is duplicated $\eta_{s(k)} \cdot \eta_{a(k)}$ times to take into account multiple pumps in each station. Furthermore, a dummy path, k_0, which starts and ends at the depot without visiting any customers, with duration $\gamma_{k_0} = 0$ and length $d_{k_0} = 0$ is generated. This path is used, in the model presented in the following, to ensure routes continuity and to force the number of routes to be lower than the number m of available AFVs. The set of all generated paths, included the duplicated ones and the dummy path, is denoted by $\tilde{K}$. All paths belonging to $\tilde{K}$ are given as input to a Path based MIP model, which is used to select paths and properly combine them to generate the routes of the optimal GVRP-CAFS solution. A coverage parameter, c_{ik}, is introduced, equal to 1 if $i \in I$ is handled in $k \in \tilde{K}$, 0 otherwise. We define by $\tilde{s}_k$ and $\tilde{a}_k$ the starting and the arrival pump of path k, respectively. If $\tilde{s}_k = 0$, it means that k starts from the depot and, similarly, $\tilde{a}_k = 0$ implies that k ends at the depot. The notation p_s is extended also to the refueling pumps $h \in \Pi$ imposing that $p_h = p_s$ where s is the AFS which h belongs to. From $\tilde{K}$, the set P of all the pairs of paths is generated. A pair $(k_1, k_2), k_1 \in \tilde{K}, k_2 \in \tilde{K}$, $k_1 \neq k_2$, exists (*compatibility rules*) if: $\tilde{a}_{k_1} = \tilde{s}_{k_2}$, the sets of customers handled in the two paths are disjoint and $t_{0\tilde{s}_{k_1}} + \gamma_{k_1} + \gamma_{k_2} + t_{\tilde{a}_{k_2}0} + p_{\tilde{s}_{k_1}} + + p_{\tilde{a}_{k_1}} + p_{\tilde{a}_{k_2}} + p^{start} \leq T_{max}$.

The following decision variables are introduced:

- Z_k, equal to 1 if $k \in \tilde{K}$ is selected, 0 otherwise;
- X_{kl}, equal to 1 if $l \in \tilde{K}$ is covered just after $k \in \tilde{K}$, 0 otherwise;
- T_k a non-negative variable representing the starting refueling time of $k \in \tilde{K} \setminus \{k_0\}$ at its final node $\tilde{a}_k$ if $\tilde{a}_k \neq 0$, the arrival time of k to the depot if $\tilde{a}_k = 0$. While, for the dummy path k_0, $T_{k_0} = p^{start}$.

The Path based MIP model for the scenario with private AFSs (first scenario of Section 1) is the following:

$$\min \sum_{k \in \tilde{K} \setminus \{k_0\}} d_k Z_k \tag{19}$$

$$\sum_{k \in \tilde{K} \setminus \{k_0\}} c_{ik} Z_k = 1 \quad \forall i \in I \tag{20}$$

$$\sum_{k_2 \in \tilde{K}} X_{k_0 k_2} \leq m \tag{21}$$

$$\sum_{\substack{k_1 \in \tilde{K}: \\ (k_1,k) \in P}} X_{k_1 k} = \sum_{\substack{k_2 \in \tilde{K}: \\ (k,k_2) \in P}} X_{k k_2} \quad \forall k \in \tilde{K} \setminus \{k_0\} \tag{22}$$

$$\sum_{\substack{k_1 \in \tilde{K}: \\ (k_1,k_2) \in P}} X_{k_1 k_2} = Z_{k_2} \quad \forall k_2 \in \tilde{K} \setminus \{k_0\} \tag{23}$$

$$T_{k_2} \geq T_{k_1} + p_{\tilde{a}_{k_1}} + \gamma_{k_2} - T_{max}(1 - X_{k_1 k_2}) \quad \forall (k_1, k_2) \in P : k_2 \neq k_0 \tag{24}$$

$$|T_{k_1} - T_{k_2}| \geq p_{\tilde{a}_{k_1}} \quad \forall k_1, k_2 \in \tilde{K} \setminus \{k_0\} : \tilde{a}_{k_1} = \tilde{a}_{k_2}, \tilde{a}_{k_1} \neq 0 \tag{25}$$

$$T_k \leq T_{max} - p_{\tilde{a}_k} + T_{max}(1 - Z_k) \quad \forall k \in \tilde{K} \setminus \{k_0\} \tag{26}$$

$$Z_k \in \{0,1\}, T_k \geq 0 \quad \forall k \in \tilde{K} \setminus \{k_0\}, \; T_{k_0} = p^{start} \tag{27}$$

$$X_{k_1 k_2} \in \{0,1\} \quad \forall (k_1, k_2) \in P \tag{28}$$

The objective function (19) concerns the minimization of the total travel distance. Each customer has to be visited exactly once (20) and the number of routes selected must not exceed the number of available AFVs (21). Constraints (22) ensure that each route is a sequence of paths where the first one and the last one is k_0. Indeed, they guarantee that each path $k \neq k_0$ selected is traveled just after a path k_1 and just before a path k_2, both selected in the same route. In fact, since constraints (22) are imposed for all paths $k \in \tilde{K} \setminus \{k_0\}$, the dummy path is the only one not requiring to have both a predecessor and a successor. Constraints (23) guarantee that a path can be inserted in a route only if it is selected. If $X_{k_1 k_2} = 1$, the starting refueling of path k_2 cannot be performed before the refueling operation of path k_1 at $\tilde{a}_{k_1}$ has been completed (24). Two AFVs cannot simultaneously refuel at the same pump (25). The refueling operation carried out after path k cannot

start before the path is completed nor finish after T_{max} (26). Constraints (27) and (28) specify variables nature.

Constraints (25) can be linearized by (29)-(30), through the introduction of the auxiliary binary variables $Y_{k_1k_2}$ which take value 1 if path k_1 reaches the AFS before k_2 and 0 otherwise.The nature of variables $Y_{k_1k_2}$ is specified by constraints (31).

$$T_{k_2} \geq T_{k_1} - 2T_{max}(1 - Y_{k_1k_2}) + p_{a_{\tilde{k}_1}} - 2T_{max}(2 - (Z_{k_1} + Z_{k_2})) \quad \forall k_1 \in \tilde{K}, k_2 \in \tilde{K} \setminus \{k_0\} : \tilde{a}_{k_1} = \tilde{a}_{k_2}, \tilde{a}_{k_1} \neq 0 \tag{29}$$

$$T_{k_1} \geq T_{k_2} - 2T_{max}Y_{k_1k_2} + p_{a_{\tilde{k}_2}} - 2T_{max}(2 - (Z_{k_1} + Z_{k_2})) \quad \forall k_1 \in \tilde{K}, k_2 \in \tilde{K} \setminus \{k_0\} : \tilde{a}_{k_1} = \tilde{a}_{k_2}, \tilde{a}_{k_1} \neq 0 \tag{30}$$

$$Y_{k_1k_2} \in \{0, 1\} \quad \forall k_1, k_2 \in P : \tilde{a}_{k_1} = \tilde{a}_{k_2}, \tilde{a}_{k_1} \neq 0 \tag{31}$$

Similarly to the Arc based formulation, the Path based model can be easily extended to the scenario with public AFSs where the refueling pumps reservation has to be addressed. Indeed, using the notation for the time windows introduced at the end of Section 4, the following constraints have to be added to the formulation (19)-(31):

$$e_{\tilde{a}_k} \leq T_k \leq l_{\tilde{a}_k} \quad \forall k \in \tilde{K} \setminus \{k_0\} \tag{32}$$

Moreover, multiple time windows associated with the pumps can be addressed cloning the pumps as explained at the end of Section 4.

6. An exact Cutting Planes approach

Cutting Planes methods (CP) belong to a broad class of algorithms, to exactly solve NP-hard optimization problems, based on the following principle. Given a MIP formulation of a problem (hereafter denoted as Original Problem - OP), a relaxation (RP) of it is obtained by omitting a class of constraints responsible to make OP hard to solve (usually, the integrality constraints) or containing a huge number of constraints. In the latter case, in fact, solving OP may become prohibitive from a computational point of view. Once the RP is solved to optimality, the feasibility of its solution is checked with respect to OP. If the optimal solution of RP is feasible for OP, then it is optimal for OP too. Otherwise, we detect the violated constraints, i.e. the *cuts*, we add them to RP and we solve it again. This procedure is

iteratively repeated until an optimal solution for OP (if any) is found. If some of the initially dropped constraints are added to RP making it infeasible, OP is also infeasible. This method is exact since it converges to an optimal solution, if it exists, in a finite number of iterations. In the worst case, all the relaxed constraints have to be added to RP before proving that either OP is infeasible or finding an optimal solution. The efficiency of the method depends on the tightness of the relaxation and on the effectiveness of the proposed cuts. Several CP variants have been proposed in the literature across the years. For a complete survey on this subject, we refer the reader to [24]. Although most of them propose to relax the integrality constraints, many others apply different approaches. For example, in [19], for the Travelling Salesman Problem, the huge number of subtours elimination constraints is relaxed.

6.1. A Cutting Planes approach for the GVRP-CAS

The proposed approach works as follows. An RP is obtained, starting from the formulation reported in Section 5, by dropping constraints (29) and (30). The obtained problem corresponds to the classical GVRP in which no limits on simultaneously refueling operations are imposed. The RP is then solved to optimality by a MIP commercial solver. The feasibility of the obtained solution is then checked with regard to the GVRP-CAFS. If it is feasible, then it is optimal for GVRP-CAFS too. Otherwise, all the violated constraints are added to RP, which is solved again. This procedure is iteratively repeated until either the solution of RP becomes feasible for GVRP-CAFS or a time limit TL is reached. In the latter case, the optimal solution of RP obtained at the last iteration is considered as the best lower bound. In fact, the RP_i, at a generic iteration i, is obtained adding further cuts to the relaxation at the previous iteration, RP_{i-1}, and, therefore, the optimal solution of RP_i, $Z(RP_i)$, must be not lower than $Z(RP_{i-1})$.

6.2. A Proactive Cutting Planes approach

We propose a modified version of the CP described in the previous section, called *CP-proactive*. In this version of the algorithm, at each iteration, given the paths selected by the optimal solution of RP, we add constraints (29) and (30) not only for the pairs of paths violating the capacity constraint, but also for all pairs of them ending at the same AFS. In this way, we add some valid inequalities as future probable cuts to prevent further violations and force the model to find a feasible schedule for the current selection of paths ending

at that AFS. If such a schedule does not exist, then, in the next iteration, the optimal solution of RP will not contain all these paths simultaneously. Such approach aims at reducing the number of iterations needed by the algorithm trying to preventively add constraints which have high probabilities to be violated in the next iterations. This approach is particularly useful when the number of visits to the same AFS s is high and the combination of the maximum route duration T_{max} and the refueling time at AFS s, p_s, are such that the visit scheduling constraints become very tight, as in the example shown in Figure 3 and described in the next subsection.

6.3. An illustrative example for CP-proactive

In this subsection, we present an example in which the CP-proactive would be very effective compared to the standard CP. In this example, T_{max} is equal to 7 hours. Moreover, there is only an AFS s which is 2 hours far from the depot. This means that, at s, refueling operations can take place between the 2^{nd} and the 5^{th} hour. Refueling time is fixed to 0.5 hours. At the first iteration, in the optimal solution of RP, we have 7 vehicles refueling at s. The arrival time of each vehicle and the related AFS occupation is illustrated in Figure 3. It is trivial to note that whichever vehicle scheduling is applied, such a solution would not become feasible for the GVRP-CAFS, because the sum of the refueling times needed (3.5 hours) is greater than the available time for refueling (3 hours, considered that 4 hours are necessary to reach the station from the depot and come back to it). The first six vehicles do not overlap, while vehicle 7 is overlapping vehicle 1. Therefore, the optimal RP solution is infeasible for the GVRP-CAFS. At the second iteration, the standard CP would add a constraint to prevent AFV 1 overlaps AFV 7. Consequently, AFV 7 arrival would be shifted but would overlap another AFV. The same would happen in cascade for the other AFVs. To detect the solution infeasibility, in terms of selected paths, the standard CP may need, in the worst case, $6! = 720$ iterations, in which at each iteration a single cut is added. The CP-proactive would add all the 720 non-overlapping constraints at the same iteration. Therefore, already from the second iteration, the optimal RP solution would not have 7 vehicles refueling at s. In this case, the convergence of the CP-proactive would be much faster compared to the standard CP approach.

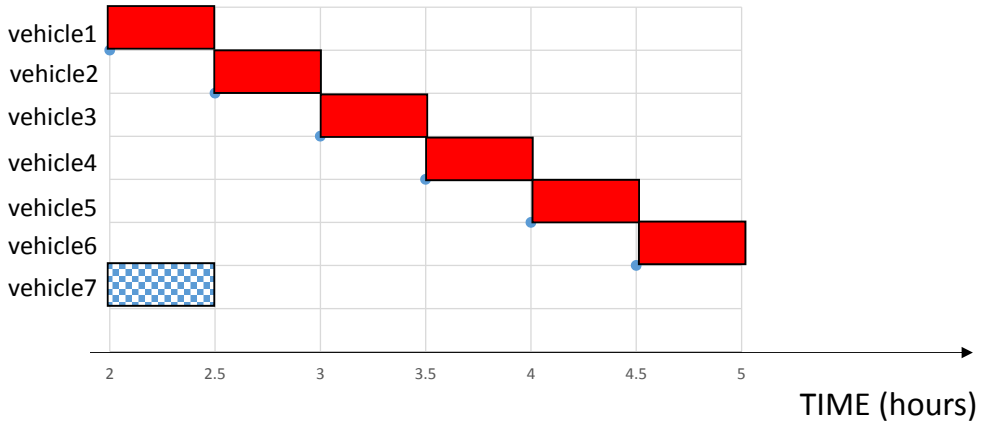

Figure 3: An illustrative example for which CP-proactive is much more efficient than CP

7. Numerical results

In this section, we test the performances of all the proposed approaches on some realistic cases with different layouts and AFS capacity tightness. In particular, we compare results provided by the two MIP models, the Arc-based MILP (A-MILP) and the Path-based MILP (P-MILP), introduced in Sections 4 and 5, respectively, and the ones detected by the two versions of the Cutting Planes method, CP and CP-proactive (introduced in Section 6), in the scenario with private AFSs (first scenario of Section 1). All tests have been carried out under Xpress 7.9 on a machine equipped with a processor Intel i7-5500U at 2.4 GHz with 16 GB of RAM, giving a CPU time limit of 3600 seconds.

Three sets of instances have been experimented, as detailed in the following.

- EMH Set: it is composed of 10 instances selected from the 40 proposed in [10]. The instances are selected if in the optimal solution of the GVRP there are at least two vehicles visiting the same AFS. The other instances are discarded because they would never be challenging for the GVRP-CAFS. The original values of the parameters have been kept, i.e., $T_{max} = 11$ hours, $p^{start} = 0.25$ hours, $p_s = 0.25$ hours $\forall s \in F$, $p_i = 0.5$ hours $\forall i \in I$, $Q = 60$ gallons and $r = 0.2$ gallons/miles

(leading to $D_{max} = 300$ miles). The number of customers varies from 6 to 20, the number of AFSs from 3 to 10 and the number of vehicles from 3 to 8. The layout of the instances is similar to that represented in Figure 1 with both the depot and the AFSs embedded in the customers area. Due to this layout, refuels may happen at very different times along the routes or do not happen at all. Therefore, these instances can be considered *slightly challenging* for the GVRP-CAFS.

- TRIANGLE Set: it is composed of 10 instances with 15 customers, 3 AFSs and 10 vehicles. The parameters have been set as $T_{max} = 11$, $p^{start} = 0$, $p_s = 0.5\ \forall s \in F$, $p_i = 0.75\ \forall i \in I$, $Q = 50$ and $r = 0.2$ (leading to $D_{max} = 250$). The layout of these instances is similar to that represented in Figure 2 with the AFSs which lay in the middle between the depot and the customers area. In this case, every vehicle needs to refuel and the refueling can be carried out only at the beginning or at the end of the route making the scheduling problem more challenging. These instances can be considered *medium challenging*.

- CENTRAL set: it is composed of 10 instances with 15 customers, only one AFS, located at the center of the customers area, and 15 vehicles. The parameters have been set as $T_{max} = 7$, $p^{start} = 0$, $p_s = 0.5\ \forall s \in F$, $p_i = 0.5\ \forall i \in I$, $Q = 50$ and $r = 0.2$ (leading again to $D_{max} = 250$). The depot is far from the customers area and the travel time from the depot to the AFS is 2 hours. In this way, the available time for refueling is small and the AFS capacity becomes a crucial issue. These instances can be considered *extremely challenging*.

In all the three sets of instances, the average speed v has been set equal to 40 miles per hour. Without loss of generality, for each AFS s, its capacity η_s is assumed to be equal to 1. Indeed, the general case in which η_s is greater than 1 can be addressed by properly introducing as many clones of s as the value of η_s. Since in the EMH instances only a refuel per route is necessary and $\eta_s = 1, \forall s \in F$, in the A-MILP we consider a number $q = m$ of clones of each fuel pump. While, since in the other two sets of instances two refuels per route are necessary, in the A-MILP we consider $q = 2m$. The results on the three sets of instances are reported respectively in Tables 2, 3 and 4. Each table is organized in the following way. The first column indicates the instance name, while the second, the third and the fourth column indicate the objective function value (i.e., the total travel distance), the solver MIP-GAP

and the required CPU time (in seconds) for the solutions obtained by the A-MILP, respectively. The next three columns report the same information for the solutions obtained by the P-MILP. The next four columns report the objective function value, the required CPU time (in seconds), the number of cuts added and the number of iterations (Iter) performed through the CP method. The next four columns report the same information for the solutions found with the CP-proactive. We emphasize in bold the optimal results. While the symbol † in the "CPU time" column indicates that the CPU time limit has been reached. Finally, the acronym NFS stands for No Feasible Solution found.

7.1. Results Analysis

Computational results obtained on the EMH set show that both the CP and the CP-proactive solve all instances to optimality within very small computational times (9.38 seconds, on average) while the P-MILP requires 216.70 seconds, on average. In particular, the average number of paths and pairs generated is about 4261.80 and 19190.30, respectively in about 8.86 seconds on average. The A-MILP is able to solve to optimality only one instance, while it reached the CPU time limit of 3600 seconds in all the other cases, with an average MIP gap of 27.34%. Both CP and CP-proactive reach the optimum at the first iteration, which means that the optimal RP solution is already feasible for OP and, therefore, no additional cuts are necessary. The larger computational times of the P-MILP can be explained by the huge number of constraints added to avoid overlapping refuels. The bad A-MILP performances, respect to the other methods, show that a path based approach is much more convenient on the GVRP-CAFS.
The same trend is confirmed on the TRIANGLE set, whose instances are more challenging. Both CP and CP-proactive require a small number of iterations (3) and number of cuts (4 and 5) to solve to optimality all instances. Computational times are slightly lower for CP-proactive, 5.87 against 7.68 seconds, but, both methods can be considered essentially equivalent in terms of performances. The P-MILP shows a good performance, solving all the instances in 26.37 seconds on average, while the A-MILP is never able to find a feasible solution within the CPU time limit. For this set of instances, the average number of paths and pairs generated is about 1048.20 and 11615.00, respectively, in about 0.41 seconds on average.

Finally, on the extremely challenging instances belonging to set CENTRAL, the CP requires, on average, 17 iterations and 17 cuts, which means

that on average, just one violated overlapping constraint is detected at each iteration. Despite the larger number of cuts added by the CP-proactive (91), it requires a smaller number of iterations (11) to converge to the optimal solution. This behavior is due to the beneficial effect of the proactive addition of cuts. Due to the high challenge level of the instances, respect to the previous set, average computational times slightly grow, 989.56 seconds for the CP and 975.15 seconds for the CP-proactive, and both methods are able to solve to optimality 8 instances over 10. The P-MILP is able to solve to optimality only 4 instances over 10 within the CPU time limit and the average computational time is 2831.34 seconds. The average number of paths and pairs generated is about 1440.20 and 12498.10, respectively, in about 1.00 seconds on average. The A-MILP is not able to detect the optimal solution in any instance and obtains an average MIP gap of 78.96%. It is worth noting that in the Total Distance columns, optimal solutions are reported in bold. While, in the cases in which the CP and CP-proactive reach the CPU time limit without proving optimality, the value reported is, actually, the best lower bound obtained.

Table 2: Comparison of methods performances on EMH instances

	A-MILP			P-MILP			CP				CP-proactive			
Instance	Total distance	GAP(%)	CPU Time	Total distance	GAP(%)	CPU Time	Total distance	CPU Time	Cuts	Iter	Total distance	CPU Time	Cuts	Iter
20c3sC5	NFS	INF	3600	**1797.49**	1797.49	303.02	**1797.49**	13.42	0	1	**1797.49**	13.42	0	1
20c3sC6	2758.14	26.16	3600	**2156.01**	2156.01	713.24	**2156.01**	1.45	0	1	**2156.01**	1.45	0	1
20c3sC7	2758.17	20.33	3600	**2758.17**	2758.17	4.64	**2758.17**	0.30	0	1	**2758.17**	0.30	0	1
20c3sC8	1393.99	0.00	3.67	**1393.99**	1393.99	0.58	**1393.99**	0.49	0	1	**1393.99**	0.49	0	1
20c3sC10	3139.72	13.02	3600	**3139.72**	3139.72	2.41	**3139.72**	0.65	0	1	**3139.72**	0.65	0	1
20c3SU1	2583.42	45.65	3600	**2583.42**	2583.42	8.30	**2583.42**	6.82	0	1	**2583.42**	6.82	0	1
S22i6s	1633.1	44.08	3600	**1633.1**	1633.1	1035.88	**1633.1**	65.58	0	1	**1633.1**	65.58	0	1
S26i6s	2431.33	42.34	3600	**2431.33**	2431.33	38.81	**2431.33**	2.31	0	1	**2431.33**	2.31	0	1
S28i6s	2158.35	15.27	3600	**2158.35**	2158.35	6.75	**2158.35**	0.81	0	1	**2158.35**	0.81	0	1
S210i6s	1585.46	39.18	3600	**1585.46**	1585.46	53.38	**1585.46**	1.95	0	1	**1585.46**	1.95	0	1
Average		27.34	3200.41	2163.70	2163.70	216.70	2163.70	9.38	0.00	1.00	2163.70	9.38	0.00	1.00

Table 3: Comparison of methods performances on TRIANGLE instances

	A-MILP			P-MILP			CP				CP-proactive			
Instance	Total distance	GAP(%)	CPU Time	Total distance	GAP(%)	CPU Time	Total distance	CPU Time	Cuts	Iter	Total distance	CPU Time	Cuts	Iter
Triangle1	NFS		3600†	**1871.61**	0.00	23.96	**1871.61**	1.04	0	1	**1871.61**	1.04	0	1
Triangle2	NFS		3600†	**2191.73**	0.00	14.42	**2191.73**	6.15	8	4	**2191.73**	4.09	6	3
Triangle3	NFS		3600†	**1872.12**	0.00	22.69	**1872.12**	6.15	5	4	**1872.12**	4.56	8	3
Triangle4	NFS		3600†	**1869.07**	0.00	30.95	**1869.07**	5.38	2	3	**1869.07**	4.52	8	3
Triangle5	NFS		3600†	**1852.73**	0.00	49.15	**1852.73**	13.28	6	6	**1852.73**	11.02	5	5
Triangle6	NFS		3600†	**1865.49**	0.00	23.31	**1865.49**	6.12	3	4	**1865.49**	5.25	4	3
Triangle7	NFS		3600†	**1898**	0.00	11.16	**1898**	2.75	4	3	**1898**	3.1	4	3
Triangle8	NFS		3600†	**2197.49**	0.00	33.02	**2197.49**	28.22	5	5	**2197.49**	13.76	7	5
Triangle9	NFS		3600†	**1862.5**	0.00	37.48	**1862.5**	3.15	1	2	**1862.5**	6.30	2	3
Triangle10	NFS		3600†	**1864.73**	0.00	17.51	**1864.73**	4.54	4	3	**1864.73**	5.04	4	3
Average				1934.55	0.00	26.37	1934.55	7.68	3.80	3.50	1934.55	5.87	5	3.20

Table 4: Comparison of methods performances on CENTRAL instances

	A-MILP			P-MILP			CP				CP-proactive			
Instance	Total distance	GAP(%)	CPU Time	Total distance	GAP(%)	CPU Time	Total distance	CPU Time	Cuts	Iter	Total distance	CPU Time	Cuts	Iter
Central1	1148.32	82.87	3600†	**953.94**	0.00	359.06	**953.94**	101.24	24	33	**953.94**	130.61	133	23
Central2	959.88	79.69	3600†	959.88	1.29	3600†	**948.69**	59.17	27	12	**948.69**	441.91	307	31
Central3	958.94	79.32	3600†	960.21	1.91	3600†	943.6	3600†	84	33	943.12	3600†	280	28
Central4	NSF		3600†	1098.89	13.55	3600†	968.2	3600†	20	72	967.96	3600†	170	17
Central5	714.55	78.43	3600†	**714.55**	0.00	106.76	**714.55**	2.74	1	2	**714.55**	2.98	3	2
Central6	845.53	78.80	3600†	845.53	4.96	3600†	**844.43**	1348.23	0	1	**844.43**	1348.23	0	1
Central7	866.66	77.52	3600†	**862.68**	2.72	3600†	**862.68**	216.52	1	2	**862.68**	148.07	3	2
Central8	712.83	76.58	3600†	**712.83**	0.00	254.10	**712.83**	8.04	2	3	**712.83**	4.60	3	2
Central9	866.86	76.37	3600†	**866.39**	0.00	1493.46	**866.39**	22.10	0	1	**866.39**	22.10	0	1
Central10	906.76	81.03	3600†	906.47	3.80	3600†	**901.19**	937.58	11	10	**901.19**	453.02	15	3
Average		78.96		888.14	2.82	2381.34	871.65	989.56	17	16.90	871.58	975.15	91	11

8. Conclusions and future works

In this paper, we introduced the GVRP-CAFS, a more realistic variant of the GVRP where the capacity of the Alternative Fuel Stations (AFS) is addressed. For this new problem, we proposed two different MILP formulations (A-MILP and P-MILP) based on arc-variables and path-variables, respectively. Moreover, we also proposed two slightly different cutting planes methods (CP and CP-proactive). Since the benchmark instances of the GVRP are not challenging for the GVRP-CAFS, because of the capacity constraint not tight, we built two real-world alike sets of 10 instances each where the capacity constraint is actually tight.

CP and CP-proactive obtain very good results within very reasonable computational times. CP-proactive is slightly faster on average, but does not systematically outperform CP. Both of them strongly outperform the P-MILP, which is significantly slower but still reaches the optimal solution in all the instances of the first two sets and on 4 instances of the last set. On the other hand, the A-MILP is not suitable to efficiently solve the GVRP-CAFS since on the two sets of instances where the capacity constraint is tight it is not able even to find a feasible solution within the CPU time limit of 1 hour.

Future works concern the development of metaheuristic approaches to address larger sized instances. Moreover, the solution approaches proposed for the GVRP-CAFS can be extended to other problems such as the VRP with Intermediate Facilities, e.g., the waste collection VRP with intermediate facilities or the VRP with Intermediate Replenishment Facilities, where the use of shared facilities among the vehicles has to be optimized (similarly to the shared use of AFSs). Another possible extension of our work is to the GVRP with multiple recharge technologies where, at each AFS, several recharging technologies are available, characterized by different recharging times and costs and the objective consists in minimizing a general cost function (given by the total travel distance and the recharging cost).

References

[1] M. Affi, H. Derbel, and B. Jarboui. Variable neighborhood search algorithm for the green vehicle routing problem. *International Journal of Industrial Engineering Computations*, 9(2):195–204, 2018.

[2] J. Andelmin and E. Bartolini. An exact algorithm for the green vehicle routing problem. *Transportation Science*, 51(4):1288–1303, 2017.

[3] T. Bektaş, E. Demir, and G. Laporte. Green vehicle routing. In *Green Transportation Logistics*, pages 243–265. Springer, 2016.

[4] M. Bruglieri, S. Mancini, F. Pezzella, and O. Pisacane. A new mathematical programming model for the green vehicle routing problem. *Electronic Notes in Discrete Mathematics*, 55:89–92, 2016.

[5] M. Bruglieri, S. Mancini, F. Pezzella, and O. Pisacane. A path-based solution approach for the green vehicle routing problem. *Computers & Operations Research*, 103:109–122, 2019.

[6] M. Bruglieri, S. Mancini, F. Pezzella, O. Pisacane, and S. Suraci. A three-phase matheuristic for the time-effective electric vehicle routing problem with partial recharges. *Electronic Notes in Discrete Mathematics*, 58:95–102, 2017.

[7] M. Bruglieri, F. Pezzella, O. Pisacane, and S. Suraci. A variable neighborhood search branching for the electric vehicle routing problem with time windows. *Electronic Notes in Discrete Mathematics*, 47:221–228, 2015.

[8] E. Demir, T. Bektaş, and G. Laporte. A review of recent research on green road freight transportation. *European Journal of Operational Research*, 237(3):775–793, 2014.

[9] G. Desaulniers, F. Errico, S. Irnich, and M. Schneider. Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, 64(6):1388–1405, 2016.

[10] S. Erdoğan and E. Miller-Hooks. A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114, 2012.

[11] Á. Felipe, M. T. Ortuño, G. Righini, and G. Tirado. A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review*, 71:111–128, 2014.

[12] W. Feng and M. A. Figliozzi. Conventional vs electric commercial vehicle fleets: A case study of economic and technological factors affecting the

competitiveness of electric commercial vehicles in the usa. *Procedia-Social and behavioral sciences*, 39:702–711, 2012.

[13] G. Hiermann, R. F. Hartl, J. Puchinger, and T. Vidal. Routing a mix of conventional, plug-in hybrid, and electric vehicles. *European Journal of Operational Research*, 2018.

[14] G. Hiermann, J. Puchinger, S. Ropke, and R. F. Hartl. The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research*, 252(3):995–1018, 2016.

[15] Y.-C. Hung and G. Michailidis. Optimal routing for electric vehicle service systems. *European Journal of Operational Research*, 247(2):515–524, 2015.

[16] J. Jemai, M. Zekri, and K. Mellouli. An nsga-ii algorithm for the green vehicle routing problem. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 37–48. Springer, 2012.

[17] A. D. Jovanović, D. S. Pamučar, and S. Pejčić-Tarle. Green vehicle routing in urban zones–a neuro-fuzzy approach. *Expert systems with applications*, 41(7):3189–3203, 2014.

[18] Ç. Koç and I. Karaoglan. The green vehicle routing problem: A heuristic based exact solution approach. *Applied Soft Computing*, 39:154–164, 2016.

[19] G. Laporte and Y. Nobert. A cutting planes algorithm for the m-salesmen problem. *Journal of the Opertional Research Society*, 31(11):1017–1023, 1980.

[20] V. Leggieri and M. Haouari. A practical solution approach for the green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 104:97–112, 2017.

[21] C. Lin, K. L. Choy, G. T. Ho, S. H. Chung, and H. Lam. Survey of green vehicle routing problem: past and future trends. *Expert Systems with Applications*, 41(4):1118–1138, 2014.

[22] S. Majidi, S.-M. Hosseini-Motlagh, S. Yaghoubi, and A. Jokar. Fuzzy green vehicle routing problem with simultaneous pickup–delivery and time windows. *RAIRO-Operations Research*, 51(4):1151–1176, 2017.

[23] S. Mancini. The hybrid vehicle routing problem. *Transportation Research Part C: Emerging Technologies*, 78:1–12, 2017.

[24] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey. Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics*, 123:397–446, 2002.

[25] A. Montoya, C. Guéret, J. E. Mendoza, and J. G. Villegas. A multi-space sampling heuristic for the green vehicle routing problem. *Transportation Research Part C: Emerging Technologies*, 70:113–128, 2016.

[26] A. Montoya, C. Guéret, J. E. Mendoza, and J. G. Villegas. The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological*, 103:87–110, 2017.

[27] M. Schneider, A. Stenger, and D. Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, 48(4):500–520, 2014.

[28] M. Schneider, A. Stenger, and J. Hof. An adaptive vns algorithm for vehicle routing problems with intermediate stops. *OR Spectrum*, 37(2):353–387, 2015.