Agile optimization for a real-time facility location problem in Internet of Vehicles networks

ORIGINAL ARTICLE

Networks

Agile Optimization for a Real-Time Facility Location Problem in Internet of Vehicles Networks

Leandro do C. Martins[1] | **Daniele Tarchi**[2] | **Angel A. Juan**[1] | **Alessandro Fusco**[2]

[1]IN3 – Computer Science Dept., Universitat Oberta de Catalunya, Barcelona, Spain

[2]Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi", University of Bologna, Viale Risorgimento 2, Bologna, Italy

Correspondence
Leandro do C. Martins, IN3, Universitat Oberta de Catalunya, 08018, Barcelona, Spain
Email: leandrocm@uoc.edu

Funding information

The Uncapacitated Facility Location Problem (UFLP) is a popular *NP-hard* optimization problem that has been traditionally applied to logistics and supply networks, where decisions are difficult to reverse. However, over the years, many new application domains of the UFLP have emerged. Some of these applications require us to re-optimize the solution quickly, as inputs change slightly but frequently over time. For instance, the 5G communication standard considers several scenarios in which real-time optimization is needed, e.g., Internet of Vehicles (IoV), Virtual Network Functions placement, and Network Controller placement. Among these, IoV scenarios take into account the presence of multiple roadside units (RSUs) that should be frequently assigned to operating vehicles. In order to ensure the desired quality of service level, the allocation process needs to be carried out frequently and efficiently, as vehicles' demands change. In this dynamic environment, the mapping of vehicles to RSUs needs to be re-optimized periodically over time. This paper proposes an *agile optimization* algorithm designed to support fast re-optimization in the described dynamic environment. The algorithm is tested using a set of existing benchmark instances, and the experiments show that it can efficiently generate high-quality and real-time results in dynamic IoV scenarios.

KEYWORDS
Uncapacitated Facility Location Problem, Internet of Vehicles, Agile Optimization, Real-time Optimization, Biased-Randomized Heuristics, Smart Cities

1 | INTRODUCTION

The Uncapacitated Facility Location problem (UFLP) [4], is a well-known *NP-hard* optimization problem in the Operations Research literature. When applied in the logistics field, it typically refers to strategic decisions in a static environment. Hence, the focus is not on providing 'good' solutions fast but on providing near-optimal ones regardless of the computational times. However, over the years many new application domains of the UFLP emerged, and some of them require the computer to frequently generate new customers-to-facilities mappings as customers' demands change over time. In this dynamic context, computing time becomes a relevant factor, since it affects our capacity to make decisions as these are required [8]. Accordingly, there is a need for 'agile' solution methodologies that can quickly provide high-quality solutions in dynamic environments. Examples of such environments can be found in the healthcare sector, where demand points, operating and distribution costs can vary over time and new facilities might be added at different time periods [1, 37, 44]. Telecommunication systems constitute one of these application areas where computing times are crucial. The 5G communication standard has defined several usage cases and applications where low latency communications need to be addressed [40]. Examples of such applications are the Internet of Vehicles (IoV) [35], the Network Functions Virtualization (NFV) [31], and the Controller Placement Problem (CPP) [41], in which decisions on *how many* facilities are needed and *where* to place them might influence the Quality of Service (QoS) level offered to the users.

The IoV can be defined as a distributed transport network that is capable of making its own decisions about driving customers to their destinations. This is achieved by having communications, storage, intelligence, and learning capabilities that allow the system to predict customers' intentions [26]. In order to properly work, the IoV network should be based on a telecommunication infrastructure able to connect every vehicle and node in the system. The vehicular communication infrastructure has been designed in the past years through several communication standards, e.g., Dedicated Short-Range Communications (DSRC) [36] and Vehicle-to-Everything (V2X) [53]. In particular, the V2X standard, developed within the 3GPP standardization body, gained lots of attention since it is based on the presence of four main communication types: Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), Vehicle-to-Network (V2N), and Vehicle-to-Pedestrian (V2P) [49, 50]. Each of these refers to a particular type of communication, and is associated with different requirements and target scenarios. Within these domains, moving cars need to exchange data with other vehicles, pedestrians, or remote nodes, either directly or through roadside units (RSUs), which are located at fixed positions on the street [35]. RSUs can be seen as the transmission nodes that allow us to interconnect any element in an IoV network. In order to properly design such an IoV system, several RSUs should be deployed to cover any vehicle, while respecting the latency requirements of the desired service [34].

Accordingly, decisions, such as the right number of RSUs to deploy and their locations, are critical in IoV systems [45]. In this context, a set of locations must be strategically selected, and RSUs have to be deployed there to provide an effective data exchanging network among moving vehicles and RSUs. Since vehicles move across the city and latency / throughput constraints must be satisfied in order to reach the desired QoS level, assigning RSUs to vehicles needs to be done fast and with a high frequency (e.g., multiple times per hour) [45]. Moreover, by efficiently selecting the RSUs that need to be activated at each time, the energy consumption of the overall system can be reduced [51].

Figure 1 represents the UFLP modeling of an IoV network. Notice that several circulating vehicles are connected to the deployed (in-service) RSUs, while other RSUs remain idle. The operative RSUs should guarantee the desired QoS level while not increasing the total deployment cost (including the energy consumption cost) more than strictly necessary.

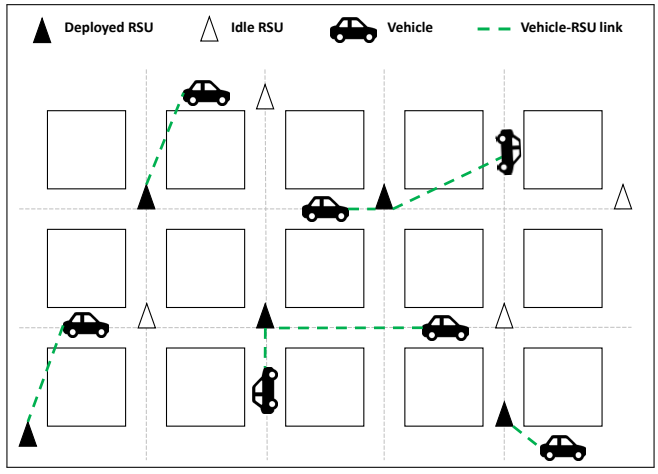


FIGURE 1 Connection of vehicles in circulation to deployed RSUs in a city.

As already commented, the vehicles-to-RSUs mapping has a critical impact on the performance of IoV systems [14], which calls for the development of novel agile methods capable of providing efficient mappings in extremely short computing times (e.g., a few seconds). In this paper, we propose a biased-randomized (BR) algorithm [46] aimed at providing solutions to the UFLP in real time. Moreover, this BR algorithm is extended into an agile optimization (AO) framework [17] in order to react and adapt to fast-changing customer demands in the system. The AO framework considers the parallel execution of the BR algorithm, so that the best among the different solutions generated by these executions is selected without increasing the wall-clock time. The UFLP has been traditionally applied to logistics and supply networks, where long-term (strategic) decisions are made. However, in dynamic environments such as virtual resource allocation, the costs associated with the activation of a resource are much lower, and decisions can be reversed in the short run. This allows us to re-optimize the location of the active resources as customer demands change over time by applying the AO framework. In the context of IoV networks, it becomes of paramount importance to select the best RSU for each vehicle. To this aim, the UFLP model allows us to effectively generate the vehicles-to-RSUs mapping that guarantees a minimum QoS level while minimizing the associated cost function.

The rest of the paper is organized as follows: Section 2 provides a literature review on related topics. Section 3 describes the UFLP from a general point of view, while Section 4 shows how we model IoV networks as an UFLP. In Section 5, the proposed AO algorithm is introduced, while the computational experiments are explained in Section 6. These computational experiments illustrate how our proposed algorithm is able to generate high-quality solutions, even for large-scale instances, in extremely short computing times. Finally, Section 7 highlights the main conclusions of this work and enumerates future lines of research.

2 | RELATED WORK

To the best of our knowledge, Balinski [4] is the first researcher who addressed and explicitly formulated the FLP as a mixed-integer programming (MIP) formulation, initially named as a simple plant location problem (SPLP). Back then, the most common techniques for solving FLPs were the exact approaches for many reasons, among others: the real-world issues and variants –such as dynamism and stochasticity– were not incorporated into these problems, and the problem instances were small or medium in size. In this regard, researchers deeply explored the use of branch-and-bound algorithms for solving the UFLP. Efroymson and Ray [19] proposed a branch-and-bound algorithm for plant location, which Khumawala improved [39]. In the latter article, the structure of the FLP was considered when designing an efficient branch-and-bound algorithm. Bilde et al. [7, 20] proposed dual-based branch-and-bound algorithms for solving the UFLP. Years later, a multi-level UFLP –where commodities are shipped from origin-level facilities to destination points via a set of intermediate-level facilities– was modeled as a mixed-integer linear program (MILP). Researchers applied two procedures for accelerating the convergence rate to optimal solutions: node simplification and primal descent procedures. More recently, de Armas et al. [16] published optimal solutions for the UFLP, where large instances were solved through a MIP model using the *Gurobi* solver (`https://www.gurobi.com/`).

Contrary to exact approaches, approximate methods do not guarantee optimality, but they can generate high-quality solutions in a reasonable amount of time –even for *NP-Hard* problems. In this regard, several heuristics and metaheuristics have been developed for solving the UFLP. Ghosh [27] introduced neighborhood search-based methods to solve the UFLP. Later, he incorporated them into tabu search (TS) frameworks. He also used complete local search with memory (CLM) algorithms to escape from local optima. Both TS and CLM methods returned costs within 0.1% of the optimal value. Another TS approach was proposed by Sun [48], where the search is guided by measuring the attractiveness of moves –the opening or closing of a facility. This attractiveness is measured by net cost changes resulting from candidate moves, which are updated –rather than re-computed– from their old values after a move has been made. When updating them, the computational time for solving the problem is highly reduced. The proposed approach generated better results than other methodologies. Resende et al. [47] developed a two-phase multi-start (MS) metaheuristic to solve the UFLP. The first stage aims at creating randomized solutions by a constructive heuristic, followed by a local search. The second step combines elite solutions with a path-relinking strategy to generate high-quality solutions. The proposed method was able to find near-optimal solutions for a large number of instances. De Armas et al. [16] proposed a fast savings-based heuristic for solving the UFLP, which is then extended into an iterated local search (ILS) metaheuristic framework. The authors generated an initial solution for the ILS through a heuristic based on closing opened facilities and re-assigning clients. Finally, an acceptance criterion is employed for accepting worse solutions within a margin limit. Apart from solving the UFLP, the authors also proposed a simheuristic algorithm [28] for solving the stochastic UFLP (SUFLP). This approach combines the proposed ILS with Monte Carlo simulation (MCS) to deal with uncertainty [12]. The authors extended a set of deterministic instances to test the simheuristic approach, which provided different efficient solutions with different trade-offs. Fernandez et al. [3] considered a video streaming application with a QoS threshold modeled as a stochastic single-allocation p-hub median problem, where a simheuristic solution is applied to select the nodes to be used. The authors study a hub-and-spoke network, where many nodes exchange real-time multimedia data modeled with a stochastic traffic behavior. Hakli and Ortakay [29] proposed an improved scatter search algorithm to solve the UFLP. The authors apply different crossover techniques to generate new solutions, and mutation operations improved the local search in the best solution. Consequently, it could overcome different population-based approaches, such as those based on swarm optimization and evolutionary algorithms. Among the solution methodologies developed and employed for solving FLPs, we highlight not only the use of exact and metaheuristic ones, but also the use of approximation algorithms [11, 13, 32, 33].

Biased-randomized algorithms [22] constitute another class of methodologies for solving combinatorial optimization problems. These methodologies extend constructive heuristics by introducing a non-uniform randomization pattern into them [24, 42]. For instance, Belloso et al. [6] use a non-uniform (skewed) probability distribution to bias a savings-based heuristic in order to solve the fleet size and mix vehicle routing problem with backhauls. The resulting biased-randomized approach proved to be competitive by generating high-quality solutions in short computing times. BR heuristics have been combined with different metaheuristic frameworks in order to efficiently solve a variety of combinatorial optimization problems, including dynamic home service routing [25], two-dimensional vehicle routing problems [18], or permutation flow-shop problems [23]. More recently, BR algorithms have been combined with parallel computing, resulting in the agile optimization strategy. Accordingly, a recent application of an AO solution method for solving a two-echelon vehicle routing problem was presented by do Carmo et al. [17], where an AO approach was able to generate feasible and high-quality solutions in real-time.

As briefly discussed before, different problems can be modeled as FLPs. However, there is a noticeable lack of literature on modeling the assignment of vehicles to RSUs as a combinatorial optimization problem in the IoV. In IoV scenarios, RSUs alongside roads are used as wireless access points, providing communication coverage to the vehicles within their coverage area [35]. Therefore, RSUs must be deployed to establish communication between both parts, i.e., RSUs and vehicles. Bozorgchenani et al. [9] propose a V2V aided approach for optimizing the task offloading selection in a urban scenario where multiple RSUs are supposedly placed along the roads. In another work, these authors extend their previous approach by considering mobile nodes that act as both relay and processing nodes for implementing on-demand vehicular services [10]. Ni et al. [45] considered an optimization model to solve the RSU deployment problem by introducing a linear programming (LP)-based clustering algorithm. This algorithm employs a utility function to measure the total benefit from the RSU deployment. The article divides the method into three steps: *(i)* RSU clustering; *(ii)* reduction to the single-node instance; and *(iii)* assigning the tasks, where *(ii)* is formulated as the single-node capacitated facility location (SNCFL) problem, and one road segment is considered to be served by multiple RSUs. Despite addressing a different issue, Khezrian et al. [38] discuss valuable information regarding the importance of establishing effective communications between RSUs and vehicles. The authors addressed a low-complexity RSU and time slot assignment in vehicular networks with multiple RSUs in tandem to minimize energy consumption. They also consider the goal of balancing the energy consumption across the RSUs. Actually, due to the limited coverage range associated with the RSUs, the downlink transmission power may strongly dominate the average power consumption of an energy-efficient RSU design. Therefore, RSUs usually prefer to communicate with nearby vehicles instead of those that are more distant. Additionally, these authors concluded that considering a subset of RSUs to communicate with vehicles gives additional flexibility in deploying data services, the functionality of which does not need to be fully replicated across all RSUs.

In our paper, we propose a new approach that is able to model the RSU deployment problem as a facility location problem. We consider several factors, including the RSU's energy consumption, the service capabilities, and the required QoS. This model is then solved using an AO method that employs a parallelized version of a BR algorithm, which allows us to generate near-optimal solutions in a fraction of time compared to other optimization approaches.

3 | THE UNCAPACITATED FACILITY LOCATION PROBLEM

As mentioned, we can model several problems from different domains as the well-known FLP. In some of these problems, no capacity limit is imposed when assigning customers to the facilities, leading to the UFLP [15]. The UFLP is defined by an undirected graph $G = (F, C, E)$, where F represents the set of facilities, C is the set of customers to be served

from any facility $i \in F$, and E is the set of edges that connect facilities with customers. An opening cost f_i characterizes each facility $i \in F$, and the cost associated with the assignment of customer j to facility i is given by $c_{ij} \geq 0$, which is usually represented by the distance from client j to the facility i, where the requested services are provided. Notice that our model assumes that the RSUs have an unlimited capacity. This assumption might not be completely realistic in all practical applications due to the generation of delays and the loss of QoS, but the relaxation allows us to validate our algorithm against the results in the literature. Let x_{ij} be the decision variable that indicates when a customer j is assigned to facility i. Therefore, $x_{ij} = 1$ if customer j is served by facility i, and $x_{ij} = 0$ otherwise. Accordingly, we use the binary variable y_i to represent when a facility i is opened ($y_i = 1$), incurring its respective opening cost f_i. Given this formulation, we can model the UFLP as:

$$\min \sum_{i \in F} \sum_{j \in C} c_{ij} x_{ij} + \sum_{i \in F} f_i y_i \tag{1}$$

subject to:

$$\sum_{i \in F} x_{ij} = 1 \qquad \forall j \in C \tag{2}$$

$$x_{ij} \leq y_j \qquad \forall i \in F, \forall j \in C \tag{3}$$

$$y_i \in \{0, 1\} \qquad \forall i \in F \tag{4}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in F, \forall j \in C \tag{5}$$

In this regard, the objective when solving the UFLP is to minimize the fixed and variable costs of the network G, given by: *(i)* the opening cost of the serving facilities; plus *(ii)* the respective serving cost from them to their customers. Equation (2) guarantees that every client is satisfied by a single facility, which must be open according to Equation (3). Finally, Equations (4) and (5) state that all decision variables are binary.

4 | UFLP FOR THE INTERNET OF VEHICLES

Although researchers have applied the classical UFLP in several contexts, they have also frequently adapted it to address specific application scenarios. This adaptation refers to alternative ways of calculating the costs for opening facilities and assigning clients to them according to the environment specifications. An IoV scenario is composed of several RSUs placed in an urban environment and acting as connection points for the vehicles moving in the area. The vehicles use the RSUs for implementing any of the previously mentioned V2X services. When selecting the best RSUs among those available in the area, a vehicle should consider its QoS requirements to choose the RSU that respects them. The goal of our approach is to consider QoS communication characteristics for developing a strategy based on UFLP. When addressing the QoS characteristics, we speak in terms of delay, throughput, and energy consumption. We aim at mapping jointly the RSUs to vehicle selection while keeping the number of active RSUs as small as possible to reduce energy consumption. With this in mind, we consider the UFLP formalization in the context of the IoV by introducing proper values for the assignment costs c_{ij} and the opening costs f_i, modeled in terms of QoS and energy consumption, respectively.

Opening Cost model

In the UFLP, the opening cost of a facility (i.e., an RSU) corresponds to the cost incurred in using that specific facility. When considering an IoV scenario, the opening cost corresponds to the cost incurred in having that RSU in particular working, i.e., its energy consumption. In the following, we consider that an RSU does not consume energy when no vehicle is allocated to it, which means that the RSU could be disconnected. On the contrary, we consider that energy consumption occurs when any of the vehicles are attached to the RSU[1]. In our model, the opening cost can be defined through an ON/OFF energy factor E_i of the RSUs. Since no composition is required, we can define:

$$f_i = E_i \tag{6}$$

Assignment Cost model

We can model the assignment cost c_{ij} as the quality of the connection between a vehicle j and an RSU i. In a real-world environment, this value is not only tied to the actual throughput S_{ij} and delay t_{ij} of the $j \rightarrow i$ connection. It is also bound to the target throughput $\bar{S}_j$ and the target delay $\bar{t}_j$ of the specific type of service requested by the vehicle j. We should keep in mind that the values S_{ij} and t_{ij} depend on: *(i)* the distance between the vehicle and the RSU; *(ii)* the propagation environment of each RSU-to-vehicle link, and *(iii)* the RSU's hardware capabilities. Furthermore, the modeling of c_{ij} should also take into consideration a minimum guaranteed level of service, specified by the variables S_j^{min} and t_j^{max}. If these constraints are not respected, the connection between i and j should have the highest cost, and we should set the corresponding c_{ij} to 1.

According to these criteria, we define two cost functions that will be later composed to obtain c_{ij}. The purpose of these cost functions is to assign each parameter (i.e., the actual throughput and delay) a value in the $[0, 1]$ range, which describes the fitness of the parameter for the user's needs. To take into account a target service value, we resort to a logistic function, whose application is often considered for modeling QoS user satisfaction in wireless communications [43]. For this objective, we have defined two logistic functions to map the throughput and the delay:

1. Throughput cost function:

$$g_1(S_{ij}) = \frac{1}{1 + e^{-\alpha_1(S_{ij} - \bar{S}_j)}}$$

2. Delay cost function:

$$g_2(t_{ij}) = \frac{1}{1 + e^{-\alpha_2(t_{ij} - \bar{t}_j)}}$$

Here, α_1 and α_2 drive the steepness of the curves and will be set to have $g_1(S_j^{min}) \approx 1$ and $g_2(t_j^{max}) \approx 1$, i.e.: the cost is maximized when the throughput is lower than the minimum, and the delay is higher than the maximum. In the model, $\bar{S}_j$ and $\bar{t}_j$ represent the points where the marginal gain is maximized. We then obtain the overall quality of service in the $[0, 1]$ range by composing the two cost functions through a weighted sum. The parameters w_1 and w_2 are designer specific variables to give more importance to either of the QoS variables:

[1] Even if RSUs usually do not go off all the way while they are ranging between the sleep and active states [30], we can consider the energy spent in a sleep state as a floor. Therefore the opening cost represents the difference between the energy consumed in the active and in the sleep states.

$$Q_{ij} = w_1 \cdot g_1(S_{ij}) + w_2 \cdot g_2(t_{ij})$$

The assignment cost c_{ij} should also consider the energy cost related to the link establishment as an additional parameter. By resorting to the Small Base Station (SBS) power-consumption model [30], we introduce E_i^{oh}. The consumed power in active mode is a function of two terms, where one is proportional to the load of the SBS. The E_i^{oh} parameter refers to the energy overhead of the i-th RSU, which is associated with the establishment of a connection –including both link setup and transmission. Since QoS and energy are semantically different, we introduce a parameter γ that allows defining the overall trade-off between operating cost and service reliability, leading to the following link-cost formulation:

$$c_{ij} = \gamma \cdot Q_{ij} + E_i^{oh}$$

Here, γ should be tuned experimentally according to the desired behavior of the algorithm.

Objective Function

Once every building block has been defined, we can rewrite the initial objective function in Equation (1) as follows:

$$\min \sum_{i \in F} \sum_{j \in C} \left(\gamma \; Q_{ij} + E_i^{oh} \right) x_{ij} + \sum_{i \in F} E_i \, y_i \tag{7}$$

5 | AGILE OPTIMIZATION SOLUTION METHOD

To solve the UFLP, we employ a BR algorithm described in Pseudocode 1. This algorithm has also been embedded into an AO framework. Being a fast algorithm, it facilitates the re-optimization of the system every time new information is incorporated into the model, which is a frequent scenario in many telecommunication networks.

5.1 | Heuristic

The proposed heuristic seeks to close open facilities according to an associated saving cost, and then it reassigns the already allocated customers to the remaining open ones. The saving cost of closing an open facility is given by: *(i)* the cost of opening it; plus *(ii)* the assignment cost of its customers; minus *(iii)* their reallocation cost to alternative facilities. Initially, all the facilities are open, and the heuristic calculates the initial saving cost for each one (line 1). The list of potential closings is sorted in descending order (line 2). Then, the top element –the one with the highest saving cost– with a positive saving value is selected, and the respective facility is closed (line 8). Since a negative saving cost implies a more expensive allocation setting than the previous one, the model automatically discards it, i.e., only positive savings are accepted (line 7). The later stages refer to the steps of: *(i)* determining the affected customers when closing the chosen facility (line 9); and *(ii)* their re-assignment to the remaining alternative opened facilities (line 10), as well as the re-computation of the individual savings given the new allocation scenario (line 11). The new saving list is then re-sorted (line 12), and the model repeats this process while the savings list is not empty (line 3). Because only a subset of affected customers should be re-allocated to alternative facilities, this process is computationally cheaper than an

opening-facilities-based strategy, where the complete set of customers must be considered in the allocation process whenever a facility is opened.

Pseudocode 1: Our Biased Randomized Algorithm

```
Data: set of facilities F, set of customers C, set of edges E, parameter β ∈ [0, 1], sol ⊆ F (initial solution as a set
      of open facilities)
L ← createSavingsList(sol);
L ← sort(L);
while L is not empty do
    Randomly select position x ∈ {1, ..., |L|} according to distribution Geom(β);
    f ← selectTheXthFacilityFromList(L);
    savingCost ← getSavingCostOfFacility(f);
    if savingCost > 0 then
        sol ← closeFacility(f);
        C′ ← getAffectedCustomers(sol);
        sol ← assignCustomersToOpenedFacilities(C′);
        L ← updateSavingsList(sol);
        L ← sort(L);
    end
    deleteFacilityFromList(f, L);
end
```

5.2 | Biased-Randomization

As the original heuristic does not incorporate any randomness to guide the search, it always generates the same solution. However, although being reasonably efficient, this strategy does not provide an efficient exploration of the solution space. To change this behavior, we incorporate a skewed probability distribution in the constructive procedure of the heuristic to extend it into a biased-randomized algorithm. This allows us to transform the deterministic heuristic into a probabilistic algorithm without losing the logic behind the original heuristic.

Following previous research work on biased-randomization techniques, we have employed the geometric probability distribution to 'induce' this biased-randomized behavior during the solution-construction process. We have chosen this distribution since it offers some convenient properties: *(i)* it only uses one parameter, β, which is easy to set since $\beta \in (0, 1)$; *(ii)* by varying the value of β, we can consider different degrees of randomness, e.g., values close to 0 emulate a uniform-random behavior, while those close to 1 emulate a greedy one; and *(iii)* generation of random variates from a geometric probability distribution is extremely fast since there are analytical expressions that allow us to obtain them –thus avoiding the use of time-expensive loops. We could also use other skewed probability distributions, although the geometric probability distribution offers a higher degree of flexibility without requiring complex fine-tuning processes. As a consequence, it is the one utilized by most of the biased-randomization algorithms published so far. Since BR techniques allow us to generate different good-quality solutions every time we execute the algorithm, we can run different threads of a BR algorithm in parallel. By doing this, we can obtain high-quality solutions with virtually the same wall-clock time as the one employed by the heuristic –which might be in the order of milliseconds. This integration of

biased-randomization techniques with parallel computing becomes an agile optimization strategy.

Pseudocode 1 describes our BR algorithm, in which a geometric distribution randomizes (smooths) the selection stage (line 5), which is greedy in the original heuristic. By considering that the sort function adopted in our algorithm is $O(n \log(n))$, we can assume that the worst-case bound for the running time is $O(n) + O(n \log(n)) = O(n \log(n))$, where n is the number of elements in the savings list.

5.3 | Agile Optimization

As described, applications such as the IoV require the real-time assignment of customers to facilities –e.g., vehicles to RSUs. These applications also require the dynamic processing of new data that is generated in dynamic systems [21]. In this regard, we need a re-optimization of the system whenever changes occur and the optimization inputs vary. This might require updating previous assignment mappings in order to include, remove, or update new vehicles' demands. In this way, the need for real-time decision-making is clear. AO takes advantage of combining powerful approaches from both parallel computing and biased-randomization of heuristics. Since BR is extremely fast in execution, easily parallelizable, flexible, and requires the fine-tuning of one single parameter –β in our case–, the combination of BR techniques with parallel computing allows us to find reasonably high-quality solutions, for a wide range of large-scale and *NP-hard* optimization problems, in real-time. In the AO framework, several independent executions of a BR algorithm –each one constituting a thread– are executed concurrently. Consequently, many different alternative solutions are generated in the same wall-clock time as the one employed by the original heuristic. Some of these solutions outperform the one proposed by the heuristic, and the best solution found is returned.

Figure 2 represents the general idea behind the AO framework, in which it runs n concurrent executions of a BR algorithm in parallel. The first execution refers to the deterministic heuristic, in which $\beta = 1$. The remaining ones are smoothed by applying $0 < \beta < 1$. In the end, the best solution found, among the several alternative solutions generated simultaneously is returned.

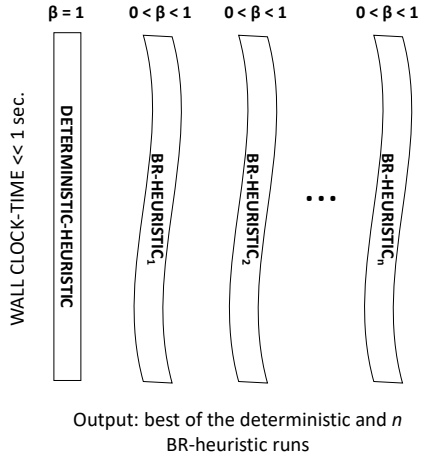


FIGURE 2 General schema of an Agile Optimization framework.

6 | COMPUTATIONAL EXPERIMENTS AND RESULTS

To test our solution approach, we used the MED benchmark instances [2] to represent the p-median problem. These instances were also used in the context of the UFLP [5]. They differ in terms of the number of customers and potential facilities: 500, 1000, 1500, 2000, 2500, and 3000. Each location is, simultaneously, a customer and a potential facility location to open. Moreover, we consider different opening cost schemes for each instance: 10, 100, and 1000 suffixes. The higher the suffix value, the lower is the opening cost of a facility. To solve these instances, we employed an Intel Core $i9$ – 9980HK processor with 32 GB of RAM. Table 1 shows the parameter settings for the computational experiments [42, 43, 52]. Notice that the opening cost values for E_i, which are derived from the number of facilities, are consistent with values considered in the literature for the energy spent for switching on an RSU [52].

TABLE 1 Problem Parameters

$\bar{S}$	$\bar{t}$	S^{min}	t^{max}	α_1	α_2	w_1	w_2	γ	E^{oh}
10 [Mb/s]	0.5 [s]	0.5 [Mb/s]	10 [s]	$1.3 \cdot 10^{-3}$	10^{-6}	0.5	0.5	{10, 30, 50}	$0.13[W \cdot s]$

6.1 | Experimental Design

As already mentioned, the power of the proposed AO approach resides in the heavy parallelization of BR algorithms, whose performance is related to its parameters. In our case, we used a geometric distribution, which is controlled by the parameter $\beta \in [0, 1]$, to introduce the random behavior in its solution space exploring process. When $\beta = 1$, the original greedy heuristic behavior is kept, while $\beta = 0$ refers to the completely random process. On the other hand, we implemented the AO framework by using a fixed number of independent threads, each one executing a BR algorithm characterized by a different beta and simulation seed value.

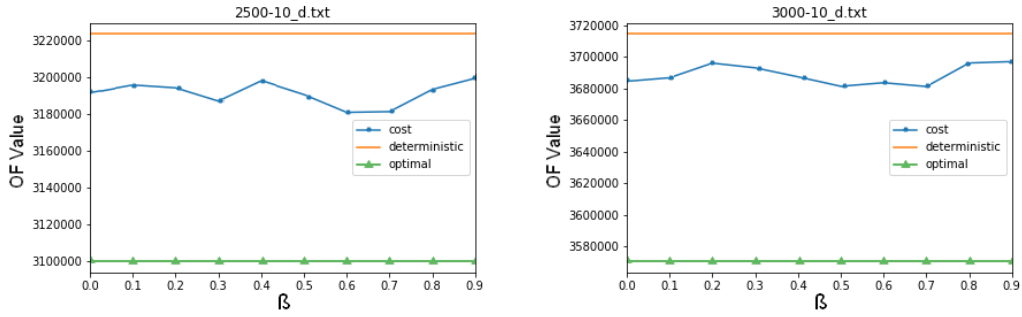
For setting β, we considered different intervals. Using steps of 0.1, and starting with 0.0 we performed 10 executions for each problem instance. Figures 3a and 3b show the convergence of the solutions over different intervals for β. These figures refer to instances $2500 - 10$ and $3000 - 10$, which are composed of 2500 and 3000 facilities, respectively. They are compared over the solution generated by the deterministic approach (orange line) and the optimal solution obtained through the Gurobi solver (green line).

As we can see, when employing intermediate values of β, it is more likely to obtain better solutions. Concretely, we notice this behavior when $\beta \in [0.4, 0.8]$. Since the remaining instances showed similar behavior, we set this interval to randomly choose the parameter β due to its capability to generate lower-cost solutions when selected in this range.

Once β is defined, we simulate the parallelization of these threads by sequentially running the biased-randomized executions –each one using a different seed for the pseudo-random number generator–, and take the slowest execution time as the wall-clock time of the overall performance. The following section presents and discusses the results for both the UFLP-IoV and UFLP problems.

6.2 | Results

To test our proposed solution method, we have set 128 available threads (parallel runs). In order to consider the case in which the heuristic is greedy ($\beta = 1$), we have added one more thread for this particular case during the execution, resulting in 129 threads. With this, we prevent the algorithm from performing worse than the original greedy



(a) Convergence of solutions of instance *2500-10*.

(b) Convergence of solutions of instance *3000-10*.

FIGURE 3 Results obtained when varying the β value from 0 to 1.

deterministic heuristic.

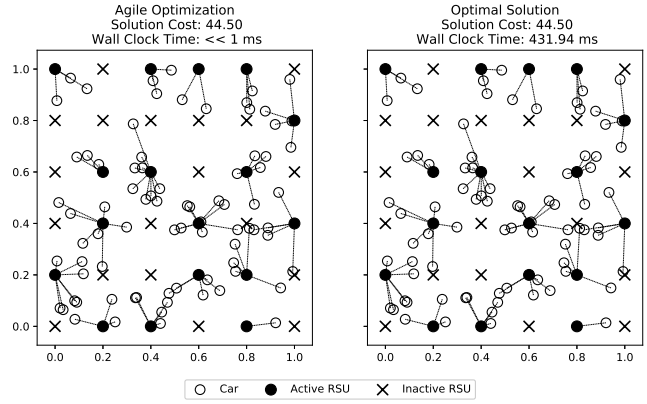
As introduced in Section 4, a new parameter, γ, must be set for the UFLP-IoV. Additionally, we have solved the corresponding model through the Gurobi optimizer to generate optimal solutions for small-sized instances and provide a proper performance comparison between the AO and Gurobi. Because of this limitation, regarding the size of the samples, three different solutions for allocating 50 cars to 16 RSUs are represented in Figure 4, which refers to assigning different weights to the QoS and energies by changing the γ value. Since we consider this problem instance of small size, the Gurobi could optimally solve it.

As we can see, our method has been able to reach the optimal solutions for the tested problem cases. In addition, we can notice that, by increasing the γ value, this solution gives more importance to the quality of the service, so there will be more active RSUs. It is important to emphasize that, by the very nature of this agile approach, the parameters can be changed for each execution, allowing greater user control over the consumption and reliability of the system. Our solution method could achieve speedups for many orders of magnitude over the Gurobi solver on the same test instances.

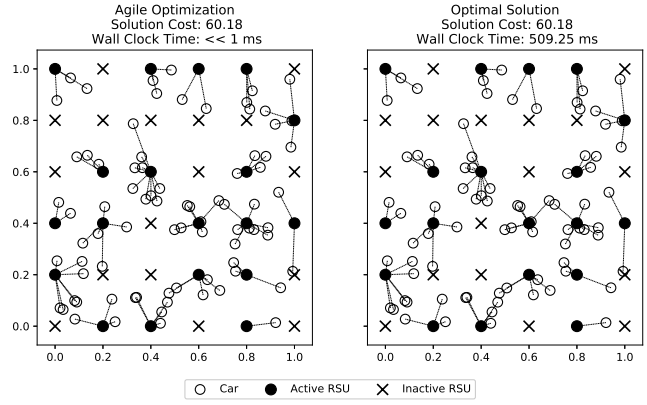
In order to have a fair comparison with other benchmarks in the literature, we will focus on a particular case of the UFLP-IoV. In this specific case, the assignment cost of vehicles to RSUs is driven by the distance between them, which mainly impacts the QoS parameters. Moreover, the energy consumption for activating an RSU corresponds to the cost of opening it, and it sets γ to 1. This simplification allows reducing the problem to the traditional UFLP. Therefore, in Table 2 shows the results obtained for the aforementioned parameter setting. For each instance, we present: *(i)* the optimal solution provided by Gurobi and its running time; *(ii)* the solution obtained with the deterministic heuristic and its running time; and *(iii)* the best solution found by our AO approach, the average solution cost, and its processing time.

When analyzing the results, we can notice that the heuristic approach can provide high-quality solutions in less than one second. By employing the AO approach, we can reduce even further the gap with respect to the optimal solutions. When comparing our method with the optimal solutions, we have an average percentage gap of 2.6%, but requiring noticeably less computational time, on the order of milliseconds. Notice that the average computational time required by Gurobi is above 4 hours, which is far beyond our practical needs when mapping vehicles to RSUs, especially since re-optimization processes might be required every few minutes.

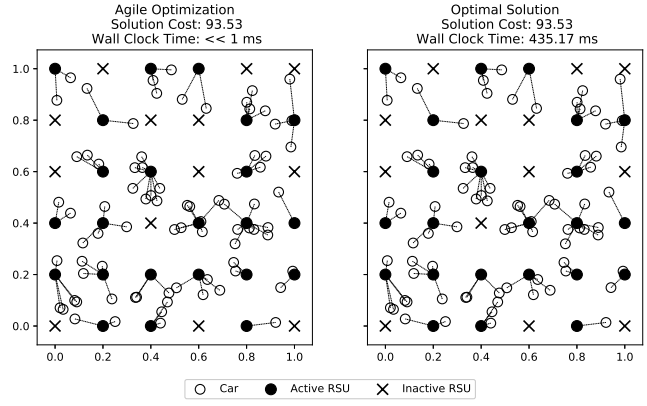
The 3GPP standardization bodies have defined several IoV services with different requirements [49, 50]. While advanced and remote driving services require very stringent latency conditions –on the order of a few milliseconds–, other services such as reporting or generic Vehicle-to-Network communications have latency requirements on the



(a) $\gamma = 10$.



(b) $\gamma = 30$.



(c) $\gamma = 50$.

FIGURE 4 The same instance is solved three times with different values of γ.

TABLE 2 Comparison of the results obtained by the proposed methodologies.

	Gurobi (1)		*Heuristic (2)*		*AO (3)*			*gap*		
Instance	*Optimal*	*Time*	*Cost*	*Time*	*Cost*	*Avg. Cost*	*Time*	*(2-1)*	*(3-1)*	*(3-2)*
500-10	798,577	15.54	816,911	0.06	811,769	825,083.54	0.06	2.3%	1.7%	-0.6%
500-100	326,790	12.69	334,182	0.01	332,657	334,433.03	0.01	2.3%	1.8%	-0.5%
500-1000	99,169	15.12	99,630	0.02	99,512	99,645.53	0.02	0.5%	0.3%	-0.1%
1000-10	1,434,154	516.45	1,496,913	0.03	1,476,219	1,496,014.28	0.07	4.4%	2.9%	-1.4%
1000-100	607,878	117.60	623,351	0.03	621,723	623,898.95	0.05	2.5%	2.3%	-0.3%
1000-1000	220,560	84.18	225,232	0.03	224,811	225,290.53	0.04	2.1%	1.9%	-0.2%
1500-10	2,000,801	12,109.82	2,081,386	0.08	2,065,163	2,092,176.60	0.14	4.0%	3.2%	-0.8%
1500-100	866,454	286.26	900,077	0.08	896,471	900,586.16	0.10	3.9%	3.5%	-0.4%
1500-1000	334,962	211.74	344,279	0.07	343,719	344,272.48	0.07	2.8%	2.6%	-0.2%
2000-10	2,558,118	2,407.97	2,683,346	0.15	2,655,188	2,687,506.16	0.21	4.9%	3.8%	-1.0%
2000-100	1,122,748	488.35	1,166,994	0.14	1,161,132	1,165,985.98	0.23	3.9%	3.4%	-0.5%
2000-1000	437,686	419.54	450,549	0.15	449,843	450,552.88	0.22	2.9%	2.8%	-0.2%
2500-10	3,099,907	240,588.51	3,223,279	0.50	3,191,427	3,231,097.21	0.50	4.0%	3.0%	-1.0%
2500-100	1,347,516	1,534.58	1,398,526	0.23	1,392,504	1,397,435.05	0.48	3.8%	3.3%	-0.4%
2500-1000	534,405	758.51	547,825	0.46	546,985	548,023.74	0.49	2.5%	2.4%	-0.2%
3000-10	3,570,766	2,960.90	3,714,590	0.61	3,684,684	3,736,407.78	0.76	4.0%	3.2%	-0.8%
3000-100	1,602,154	8,873.80	1,653,616	0.72	1,649,793	1,654,669.25	0.72	3.2%	3.0%	-0.2%
3000-1000	643,463	1,692.01	660,541	0.69	659,298	660,499.33	0.69	2.7%	2.5%	-0.2%
Average	1,200,339	15,171.87	1,245,624	0.23	1,236,828	1,248,532.14	0.27	3.1%	2.6%	-0.5%

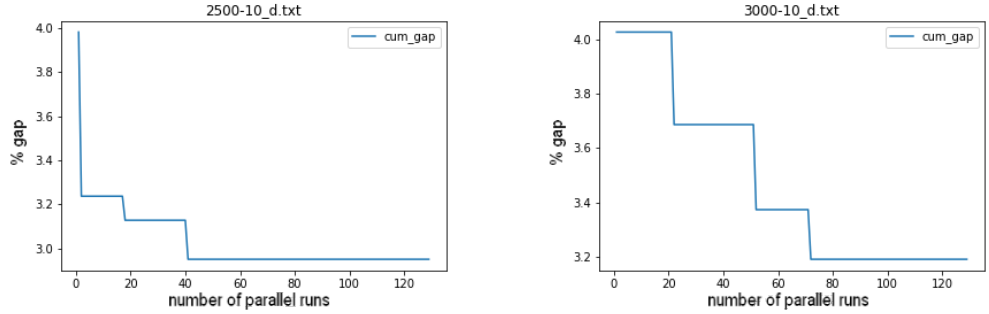
order of seconds. Hence, low latency services are preferred to be mapped on small instances, whereas we can map services with no stringent latency on bigger instances (i.e., managed as large populations).

In Figure 5, we present the convergence of the best-found solutions according to the increase in the number of threads. As mentioned, we base our method on multiple executions of a BR algorithm. By plotting the cumulative minimum of the solutions given by these threads, we can determine a good trade-off between the number of threads required and the overall improvement over the deterministic solution. In Figures 5a and 5b, this convergence is presented for problem instances $2500-10$ and $3000-10$, respectively. Since we base the proposed algorithm on an efficient heuristic, its embedding into a BR algorithm was also able to produce high-quality solutions. As expected, the more threads are employed, the higher the quality of the resulting solution.

Figure 6 depicts the improvement rate of the 129 generated solutions during our AO life-cycle in terms of the gap. In this case, we compare our AO results with the heuristic ones. As we can see, for problem instance $1000-10$ (Figure 6a), about 57% of the generated solutions are better than the result of the deterministic heuristic.

7 | CONCLUSIONS

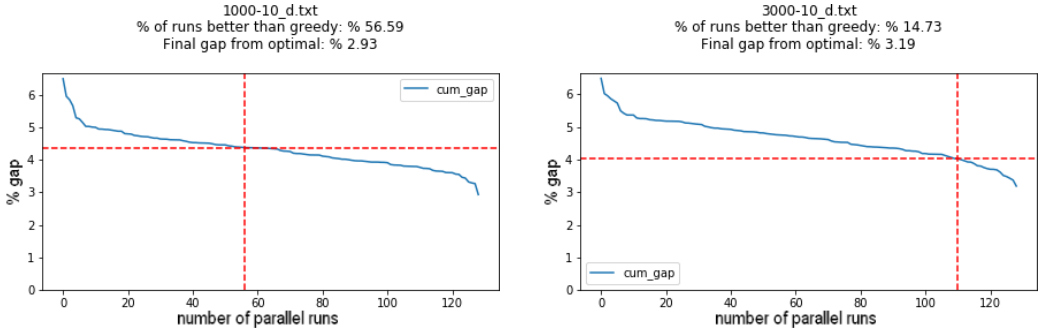
In this paper, an AO framework based on the composition of multiple biased-randomized runs of a constructive heuristic has been developed for solving the UFLP. Unlike the logistics and supply networks, where decisions are difficult to reverse and time is not a constraint for generating feasible solutions, we approach the UFLP in the context of 5G systems. In this case, the dynamic environment must be re-optimized as customer demands change over time. For doing that,



(a) Convergence of solutions of instance *2500-10*.

(b) Convergence of solutions of instance *3000-10*.

FIGURE 5 Results obtained when varying the number of parallel runs.



(a) Convergence of solutions of instance *1000-10*.

(b) Convergence of solutions of instance *3000-10*.

FIGURE 6 Results obtained when varying the number of parallel runs, including the improvement rate of the solutions when comparing with [16]'s heuristic, given by the red cross intersection.

we have extended an existing heuristic by introducing a random component into its solution space exploring process, controlled by a single parameter β. The resulting biased-randomized heuristic is able to produce different solutions that can be better than the original one by regulating the *greediness* of the algorithm and tuning a factor $\beta \in [0, 1]$. By exploiting the high parallelization capabilities of modern hardware, we can perform multiple independent runs without varying the efficient wall-clock time of the computation. Therefore, in the next step, this extended heuristic has been embedded into the AO framework to run several executions in parallel, thus requiring no additional wall-clock time.

As the results show, our method is able to obtain solutions that are virtually equivalent to those generated by a commercial solver for the UFLP-IoV. While the solver employs many hours, our approach is able to generate the solutions in less than a second. Regarding the use of our AO framework for solving this class of dynamic problems in a real-life scenario, the use of graphical processing units (GPU) could be employed to speed up computational times even further. This is, in fact, one of the future research lines we would like to explore. Another one would be considering a multi-period horizon, so we can test our methodology in combination with forecasting techniques that help to predict the future evolution of vehicles' demands.

ACKNOWLEDGMENTS

This research was partially supported by the Spanish Ministry of Science (PID2019-111100RB-C21 /AEI/ 10.13039/501100011033, RED2018-102642-T) and the Erasmus+ program (2019-I-ES01-KA103-062602). A major part of this work was done during A. Fusco's visit to the Universitat Oberta de Catalunya, Spain, supported by the Erasmus+ Study program of the European Union. We also thank Jon Raleigh for the final review of the paper.

REFERENCES

[1] A. Afshar and A. Haghani, *Modeling integrated supply chain logistics in real-time large-scale disaster relief operations*, Socio-Economic Planning Sci. **46** (2012), 327–338.

[2] S. Ahn, C. Cooper, G. Cornuéjols, and A. Frieze, *Probabilistic analysis of a relaxation for the k-median problem*, Math. Oper. Res. **13** (1988), 1–31.

[3] S. Alvarez Fernandez, D. Ferone, A.A. Juan, and D. Tarchi, *A simheuristic algorithm for video streaming flows optimisation with qos threshold modelled as a stochastic single-allocation p-hub median problem*, J. Simulation (2021), 1–14.

[4] M.L. Balinski, *On finding integer solutions to linear programs*, Proceedings of IBM Scientific Symposium on Combinatorial Problems, IBM White Plains, NY, USA, 1966, pp. 225–248.

[5] F. Barahona and F.A. Chudak, *Near-optimal solutions to large-scale facility location problems*, Discr. Optim. **2** (2005), 35–50.

[6] J. Belloso, A.A. Juan, and J. Faulin, *An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with backhauls*, Int. Trans. Oper. Res. **26** (2019), 289–301.

[7] O. Bilde and J. Krarup, "*Sharp lower bounds and efficient algorithms for the simple plant location problem*," *Studies in Integer Programming*, P. Hammer, E. Johnson, B. Korte, and G. Nemhauser (eds.), Elsevier, Amsterdam, Netherlands, 1977, pp. 79–97.

[8] C. Boonmee, M. Arimura, and T. Asada, *Facility location optimization model for emergency humanitarian logistics*, Int. J. Disaster Risk Reduction **24** (2017), 485–498.

[9] A. Bozorgchenani, D. Tarchi, and G.E. Corazza, *A control and data plane split approach for partial offloading in mobile fog networks*, 2018 IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 2018.

[10] A. Bozorgchenani, D. Tarchi, and G.E. Corazza, *Mobile edge computing partial offloading techniques for mobile urban scenarios*, 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 2018.

[11] M. Charikar and S. Guha, *Improved combinatorial algorithms for the facility location and k-median problems*, 40th Annual Symposium on Foundations of Computer Science, New York City, NY, USA, Oct. 1999, pp. 378–388.

[12] M. Chica, A.A. Juan, C. Bayliss, O. Cordón, and W.D. Kelton, *Why simheuristics? benefits, limitations, and best practices when combining metaheuristics with simulation*, SORT-Statistics Oper. Res. Trans. (2020), 311–334.

[13] F.A. Chudak and D.B. Shmoys, *Improved approximation algorithms for the uncapacitated facility location problem*, SIAM J. Comput. **33** (2003), 1–25.

[14] R. Cohen, L. Lewin-Eytan, J.S. Naor, and D. Raz, *Near optimal placement of virtual network functions*, 2015 IEEE Conference on Computer Communications (INFOCOM), Kowloon, Hong Kong, 2015, pp. 1346–1354.

[15] G. Cornuéjols, G.L. Nemhauser, and L.A. Wolsey, *The uncapacitated facility location problem*, Tech. report 605, Cornell University, Ithaca, NY, USA, 1983.

[16] J. de Armas, A.A. Juan, J.M. Marquès, and J.P. Pedroso, *Solving the deterministic and stochastic uncapacitated facility location problem: From a heuristic to a simheuristic*, J. Oper. Res. Soc. **68** (Oct. 2017), 1161–1176.

[17] L. do C. Martins, P. Hirsch, and A.A. Juan, *Agile optimization of a two-echelon vehicle routing problem with pickup and delivery*, Int. Trans. Oper. Res. **28** (2021), 201–221.

[18] O. Dominguez, A.A. Juan, I.A. De La Nuez, and D. Ouelhadj, *An ILS-biased randomization algorithm for the two-dimensional loading HFVRP with sequential loading and items rotation*, J. Oper. Res. Soc. **67** (2016), 37–53.

[19] M.A. Efroymson and T.L. Ray, *A branch-bound algorithm for plant location*, Oper. Res. **14** (1966), 361–368.

[20] D. Erlenkotter, *A dual-based procedure for uncapacitated facility location*, Oper. Res. **26** (1978), 992–1009.

[21] L. Fan, G. Li, J. Yang, H. Sun, and Z. Luo, *The design of IOV overall structure and the research of IOV key technology in intelligent transportation system*, Proceedings of the 2015 International Power, Electronics and Materials Engineering Conference, Atlantis Press, 2015, pp. 1079–1084.

[22] D. Ferone, A. Gruler, P. Festa, and A.A. Juan, *Enhancing and extending the classical GRASP framework with biased randomisation and simulation*, J. Oper. Res. Soc. **70** (2019), 1362–1375.

[23] D. Ferone, S. Hatami, E.M. González-Neira, A.A. Juan, and P. Festa, *A biased-randomized iterated local search for the distributed assembly permutation flow-shop problem*, Int. Trans. Oper. Res. **27** (2020), 1368–1391.

[24] A. Ferrer, D. Guimarans, H. Ramalhinho, and A.A. Juan, *A BRILS metaheuristic for non-smooth flow-shop problems with failure-risk costs*, Expert Syst. with Appl. **44** (2016), 177–186.

[25] C. Fikar, A.A. Juan, E. Martinez, and P. Hirsch, *A discrete-event driven metaheuristic for dynamic home service routing with synchronised trip sharing*, Eur. J. Indus. Eng. **10** (2016), 323–340.

[26] M. Gerla, E.K. Lee, G. Pau, and U. Lee, *Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds*, 2014 IEEE World Forum on Internet of Things (WF-IoT), 2014, pp. 241–246.

[27] D. Ghosh, *Neighborhood search heuristics for the uncapacitated facility location problem*, Eur. J. Oper. Res. **150** (2003), 150–162.

[28] A. Gruler, J. Panadero, J. de Armas, J.A. Moreno, and A.A. Juan, *A variable neighborhood search simheuristic for the multi-period inventory routing problem with stochastic demands*, Int. Trans. Oper. Res. **27** (2020), 314–335.

[29] H. Hakli and Z. Ortacay, *An improved scatter search algorithm for the uncapacitated facility location problem*, Comput. Indus. Eng. **135** (2019), 855–867.

[30] F. Han, S. Zhao, L. Zhang, and J. Wu, *Survey of strategies for switching off base stations in heterogeneous networks for greener 5G systems*, IEEE Access **4** (2016), 4959–4973.

[31] J.G. Herrera and J.F. Botero, *Resource allocation in NFV: A comprehensive survey*, IEEE Trans. Network Service Manage. **13** (2016), 518–532.

[32] K. Jain, M. Mahdian, and A. Saberi, *A new greedy approach for facility location problems*, Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing, Montreal, Quebec, Canada, 2002, pp. 731–740.

[33] K. Jain and V.V. Vazirani, *Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation*, J ACM **48** (2001), 274–296.

[34] B. Ji, X. Zhang, S. Mumtaz, C. Han, C. Li, H. Wen, and D. Wang, *Survey on the internet of vehicles: Network architectures and applications*, IEEE Commun. Standards Magazine **4** (2020), 34–41.

[35] O. Kaiwartya, A.H. Abdullah, Y. Cao, A. Altameem, M. Prasad, C.T. Lin, and X. Liu, *Internet of vehicles: Motivation, layered architecture, network model, challenges, and future aspects*, IEEE Access **4** (2016), 5356–5373.

[36] J.B. Kenney, *Dedicated short-range communications (DSRC) standards in the United States*, Proc. IEEE **99** (2011), 1162–1182.

[37] D. Khayal, R. Pradhananga, S. Pokharel, and F. Mutlu, *A model for planning locations of temporary distribution facilities for emergency response*, Socio-Economic Planning Sci. **52** (2015), 22–30.

[38] A. Khezrian, T.D. Todd, G. Karakostas, and M. Azimifar, *Energy-efficient scheduling in green vehicular infrastructure with multiple roadside units*, IEEE Trans. Vehicular Technology **64** (2014), 1942–1957.

[39] B.M. Khumawala, *An efficient branch and bound algorithm for the warehouse location problem*, Manage. Sci. **18** (1972), B–718.

[40] M.A. Lema, A. Laya, T. Mahmoodi, M. Cuevas, J. Sachs, J. Markendahl, and M. Dohler, *Business case and technology analysis for 5G low latency applications*, IEEE Access **5** (2017), 5917–5935.

[41] J. Lu, Z. Zhang, T. Hu, P. Yi, and J. Lan, *A survey of controller placement problem in software-defined networking*, IEEE Access **7** (2019), 24290–24307.

[42] D. Mazza, A. Pagès-Bernaus, D. Tarchi, A.A. Juan, and G.E. Corazza, *Supporting mobile cloud computing in smart cities via randomized algorithms*, IEEE Syst. J. **12** (2018), 1598–1609.

[43] D. Mazza, D. Tarchi, and G.E. Corazza, *A user-satisfaction based offloading technique for smart city applications*, 2014 IEEE Global Communications Conference, Austin, TX, USA, 2014, pp. 2783–2788.

[44] M. Moeini, Z. Jemai, and E. Sahin, *Location and relocation problems in the context of the emergency medical service systems: A case study*, Central Eur. J. Oper. Res. **23** (2015), 641–658.

[45] Y. Ni, J. He, L. Cai, J. Pan, and Y. Bo, *Joint roadside unit deployment and service task assignment for internet of vehicles (IoV)*, IEEE Internet Things J. **6** (2018), 3271–3283.

[46] C.L. Quintero-Araujo, J.P. Caballero-Villalobos, A.A. Juan, and J.R. Montoya-Torres, *A biased-randomized metaheuristic for the capacitated location routing problem*, Int. Trans. Oper. Res. **24** (2017), 1079–1098.

[47] M.G. Resende and R.F. Werneck, *A hybrid multistart heuristic for the uncapacitated facility location problem*, Eur. J. Oper. Res. **174** (2006), 54–68.

[48] M. Sun, *Solving the uncapacitated facility location problem using tabu search*, Comput. Oper. Res. **33** (2006), 2563–2589.

[49] Technical Specification Group Services and System Aspects, *Enhancement of 3GPP support for V2X scenarios; Stage 1 (Release 16)*, Technical specification 3GPP TS 22.186 v16.2.0, 3rd Generation Partnership Project, 2019.

[50] Technical Specification Group Services and System Aspects, *Service requirements for V2X services; Stage 1 (Release 16)*, Technical specification 3GPP TS 22.185 v16.0.0, 3rd Generation Partnership Project, 2020.

[51] Q. Wei, L. Wang, and A. Fei, *Energy minimization for infrastructure-to-vehicle communications with multiple roadside units*, 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), Xian, China, 2019.

[52] G. Yu, Q. Chen, and R. Yin, *Dual-threshold sleep mode control scheme for small cells*, IET Commun. **8** (2014), 2008–2016.

[53] H. Zhou, W. Xu, J. Chen, and W. Wang, *Evolutionary V2X technologies toward the internet of vehicles: Challenges and opportunities*, Proc. IEEE **108** (2020), 308–323.