A game theory-based route planning approach for automated vehicle collection

Mohamed Hadded, Pascale Minet, Jean-Marc Lasgouttes

A Game Theory-based Route Planning Approach for Automated Vehicle Collection

Mohamed Hadded[*†] Pascale Minet[‡] Jean-Marc Lasgouttes[*]

March 3, 2021

Abstract

We consider a shared transportation system in an urban environment where human drivers collect vehicles that are no longer being used. Each driver, also called a platoon leader, is in charge of driving collected vehicles as a platoon to bring them back to some given location (e.g. an airport, a railway station). Platoon allocation and route planning for picking up and returning automated vehicles is one of the major issues of shared transportation systems that need to be addressed. In this paper, we propose a coalition game approach to compute 1) the allocation of unused vehicles to a minimal number of platoons, 2) the optimized tour of each platoon and 3) the minimum energy consumed to collect all these vehicles. In this coalition game, the players are the parked vehicles, and the coalitions are the platoons that are formed. This game, where each player joins the coalition that maximizes its payoff, converges to a stable solution. The quality of the solution obtained is evaluated with regard to three optimization criteria and its complexity is measured by the computation time required. Simulation experiments are carried out in various configurations. They show that this approach is very efficient to solve the multi-objective optimization problem considered, since it provides the optimal number of platoons in less than a second for 300 vehicles to be collected, and considerably outperforms other well-known optimization approaches like MOPSO (Multi-Objective Particle Swarm Optimization) and NSGA-II (Non dominated Sorting Genetic Algorithm).

1 Introduction

Road traffic is considered to be responsible for a third of all polluting gas emissions [1]. The dispersion and the increase in the concentration of pollutant levels in the air have adverse effects on the environment and human health. Environmentally-friendly cities tend to promote shared transportation systems

[*]Inria Paris, RITS Project-team
[†]Vedecom, REVECOM Team, mohamed.elhadad@vedecom.fr
[‡]Inria Paris, EVA Project-team

like electric carsharing services [2] or car rental, which have been designed for people who do not use a vehicle regularly but wish to use one occasionally. There exist two types of rental services for electric vehicles: one-way and two-way systems [3]. In two-way (or round-trip) systems, the user returns the vehicle to the rental station where it was originally rented. One-way systems provide more freedom to the user, who may leave the vehicle where he/she wants, provided the location has been indicated to the rental station. As a consequence, users may leave the rented vehicles at any parking place, regular or dedicated to electric vehicle charging or carsharing. These unused vehicles must then be collected and returned to a rental station. This is the problem we address in this paper.

Among shared transportation systems, one-way sharing systems like SHARE NOW[1] or Mobilib[2] are favored by users because they allow them to leave rented vehicles anywhere in the urban network. However, the management of such a service requires more effort from the service provider, in order to maintain a distribution of vehicles adapted to demand. In the long term, fully autonomous vehicles seem to be the best approach to this redistribution task. However, due to technical and legal concerns, this is only a long-term solution, and less ambitious ones have to be explored in the meantime. Asking individual professional drivers to collect the vehicles and to redistribute them is an expensive and non-optimal procedure. The intermediate solution is to adopt a human supervised mode, but with as little human intervention as possible. This is possible if the human is guiding several vehicles at a time. A first idea is to use trucks [4] to collect the unused vehicles. While small trucks are used successfully in most bike-sharing systems, the use of bigger trucks in city centers raises acceptability issues, at least in Europe. Another solution is to rely on towing, where a human-driven leader vehicle pulls several empty ones. This towing can either be physical (like the stackable vehicles of the European ESPRIT project[3]) or virtual, in which case the term "platooning" is preferred.

The French ANR VALET project[4], based on earlier pioneering work [5, 6], investigated this platooning solution, with unused vehicles having enough automation capabilities to be collected and returned to the rental station in an *urban platoon*: one human-driven vehicle is followed by typically up to 5 vehicles which track it automatically. The technology required to form a platoon is the same as for Cooperative Adaptive Cruise Control (CACC) [7]. It relies on sensors (lidar, camera…) for environment perception, and multiple wireless interfaces, which enable the vehicles to communicate with each other and with the infrastructure. The number of platoons required to collect all the parked vehicles should be minimized for cost-efficiency reasons. The longest platoon tour duration should be minimized for time-efficiency reasons. In addition, the

[1] `https://www.share-now.com/`.

[2] *Paris launches new Mobilib car-sharing service*, `https://newmobility.news/2019/05/07/paris-launches-new-mobilib-car-sharing-service/`, May 2019.

[3] *H2020 ESPRIT, Easily diStributed Personal RapId Transit*, `https://cordis.europa.eu/project/id/653395`.

[4] *ANR VALET project: Automatic redistribution of carsharing vehicles and realization of a parking Valet*, `https://anr.fr/Project-ANR-15-CE22-0013`.

total energy consumed by all the platoons should also be minimized for energy-efficiency reasons. That is why each platoon should follow an optimized route to pick up the parked vehicles and return to the rental station. However, several constraints must be taken into account. The residual energy of the vehicles in the platoon should not be depleted before they arrive back at the rental station. The number of vehicles in a platoon is limited for safety reasons. Each platoon leader starts and ends his/her tour at the rental station. Each parked vehicle is picked up by exactly one platoon leader. Together, these criteria and constraints form the PROPAV problem (Platoon Route Optimization for Picking up Automated Vehicles) [8].

For the sake of simplicity, we will assume here that all vehicles have to be returned to the same station. This is usually true for rental cars, and also for car-sharing when vehicles have to be serviced. There are undoubtedly many variations to this problem: several rental stations instead of one, choice of reallocation strategy, different performance criteria. These questions will become natural when one goes beyond the proof-of-concept stage and designs a real-world service with an industrial partner. For now, our intent is to study a reference problem that can be extended later; our belief is that our algorithm and our findings will remain valid with few adaptations.

The goal of this paper is to show how to use game theory to define the problem considered and find 1) the optimized number of platoons needed, 2) the optimized tour followed by each platoon leader, while 3) minimizing the total energy consumed. We start with a short overview of related work. Section 3 shows how a coalition game can model the vehicle collection by platoons. Section 4 reports experimental results obtained in an extensive performance evaluation. The results obtained by the coalition game are compared with those obtained by well-known optimization heuristics, namely MOPSO and NSGA-II. Finally, Section 6 summarizes the main contributions of the paper and presents future directions.

2 Related work

Independently of the technique used to relocate vehicles, an efficient algorithm is essential in terms of monetary cost, but also occupation of the public road network. A Tabu search has been proposed to optimize the tours of trucks in charge of picking up unused vehicles [4]. This approach has been extended by using three heuristics to minimize the time needed to collect and return the vehicles to the rental station [9]. Another approach is to consider two objectives: minimizing the staff required and the number of shared vehicles necessary to meet user demand [10]. The electric vehicle relocation problem can also be formulated as a pickup and delivery problem [11]. This requires that, on the one hand, each tour starts and ends at the rental station and, on the other hand, each tour has a duration that is less than or equal to an acceptable threshold.

As part of the assessment of VALET, the PROPAV problem has been specified as an integer linear programming problem [8], and then solved for various

configurations, using NSGA-II (Non-dominated Sorting Genetic Algorithm II) [12].

An earlier work used a genetic algorithm to solve the Vehicle Routing Problem (VRP), which has similarities to the PROPAV problem [13]: the vehicles in the VRP stand for the platoon leaders, whereas the customers waiting to be served stand for the parked vehicles to collect. However, the objectives to optimize and the constraints to meet differ.

In this paper, we use a totally different framework based on game theory. More precisely, the PROPAV problem is modeled as a coalition formation game [14]. Each final coalition represents a platoon. The number of coalitions obtained is equal to the number of human drivers required to collect the unused vehicles. In order to prove the efficiency of the proposed game theory-based optimization framework, we compare the simulation results obtained with those obtained previously [8] with MOPSO (Multi-Objective Particle Swarm Optimization) [15] and NSGA-II.

Coalition game theory has been used in multiple applications. An example is wireless communication networks [14], where some wireless agents are used to collect data from tasks arbitrarily located and transmit them to a centralized wireless receiver. In this problem, modeled as a coalition game, each coalition consists of some agents that move between the different tasks present in the coalition to collect their data. The agents join a coalition to improve their benefit by means of a higher throughput and a smaller latency. Simulation results highlight a benefit greater than 30% for a network with 5 agents and 25 tasks compared to a fair allocation of the tasks to the closest agents.

Another application is concerned with Content Delivery Networks (CDNs) [16], where video content items are organized into clusters having similar request profiles. The popularity of each cluster member can be deduced from the popularity of its cluster representative, which allows a very strong decrease in the number of predictors used to predict the popularity of video content in the next time interval, based on its popularity in the past. The main goal is to greatly improve the Quality of Service perceived by the user (i.e. a shorter delay to provide the video content requested) by storing the most popular content items close to end users. Using traces of a real CDN, simulation results show that the coalition game provides a much more accurate clustering than the K-means algorithm. Furthermore, the time required by K-means is much greater than that of the coalition game (e.g. more than 3 times for 50 video content items).

The very good results obtained by the coalition game with respect to the state-of-the-art entice us to apply the coalition game to the PROPAV problem. The results can then be assessed by comparison with NSGA-II and MOPSO, both in terms of quality of the solution and of computation time.

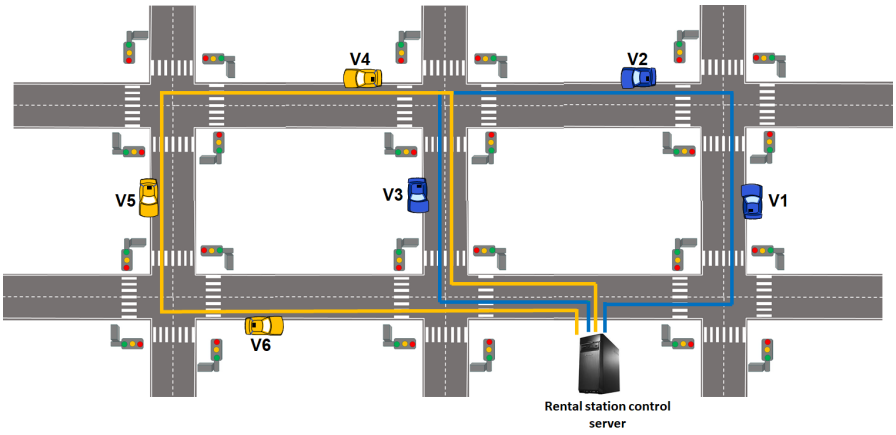

Figure 1: Six vehicles collected by two platoon leaders: a yellow one and a blue one.

3 Problem statement

3.1 Intuitive idea

Given a set of automated electric vehicles parked at various locations in a city and localized by their GPS positions, one rental station and a set of human drivers, the PROPAV problem consists in computing, on the one hand, the minimum number of human drivers needed to pick up these parked vehicles, drive them in platoon (one platoon per human driver) back to the rental station and, on the other hand, the optimized tours to be followed by the human drivers.

In the example depicted in Figure 1, two human drivers are needed to drive the unused vehicles back to the parking lot of the rental station. The optimized tours are represented in blue (v_1, v_2 and v_3) and yellow (v_4, v_5 and v_6).

Let D be the number of available platoon drivers. Intuitively, the PROPAV problem consists in designing the tour of each of the K platoons, with $1 \leq K \leq D$, such that 1) the longest platoon tour duration, 2) the number of platoons, and 3) the total energy consumed by the platoon leaders and the vehicles that are picked up are minimized. The constraints to take into account are: 1) each vehicle in the platoon should have enough energy to get back to the rental station while following the platoon driver (i.e. the tour duration should meet the energy constraint of the vehicles in the platoon which includes both the leader and its followers); 2) the rental station is the starting point and the ending point of each tour; 3) each parked vehicle should be picked up.

3.2 PROPAV as an optimization problem

Before defining the PROPAV problem as a multi-objective optimization problem, we introduce in Table 1 the notation adopted in this paper. The following

Table 1: Notation.

Notation	Meaning
D	the number of human drivers.
N	the number of parked vehicles to collect.
o	the rental station.
d	a driver $\in [1, D]$.
l	a location to visit to collect a parked vehicle $\in [1, N]$.
v	a vehicle $\in [1, N] \cup [1, D]$, either platoon leader or follower.
s	the average speed of a vehicle in a platoon.
d_{ij}	the distance between i and j.
pow	the power consumed by any vehicle.
res_v	the residual energy of vehicle $v \in [1, N] \cup [1, D]$, leader or follower in its platoon.
Q	the maximum number of vehicles per platoon.
m_{ij}^d	a binary decision variable equal to 1 if driver $d \in [1, D]$ moves directly from location i to location j, and 0 otherwise.
$prec_{ij}^d$	a binary variable equal to 1 if driver $d \in [1, D]$ has already visited location i when he/she visits location j, and 0 otherwise.

assumptions are made in this paper:

- At each location, there is only one parked vehicle. Hence, the number of vehicles to collect is equal to the number of locations to visit. If there are several vehicles at a location, then the location can be duplicated.
- The driving vehicle (leader) and its following vehicles consume the same amount of energy to travel the same distance.
- All the vehicles (i.e. the leader and the followers) in a platoon require the same amount of power and travel at the same average speed. It has been observed that most electric vehicles cover 100 km over 10 kWh without taking into account air conditioning, heating, etc.
- The energy consumed by a vehicle is equal to the vehicle power times the distance traveled, divided by the average speed.

The platoon route optimization for collecting parking vehicles can be modeled as a multi-objective minimization problem with three objectives, subject to some constraints, with the notations introduced in Table 1.

The PROPAV problem: Minimize:

- the maximum tour length

$$\min \max_{d\in[1,D]} \sum_{i=0}^{N}\sum_{j=0}^{N} m_{ij}^{d}\ d_{ij}, \tag{1}$$

- the number of platoons

$$\min \sum_{d=1}^{D}\sum_{j=1}^{N} m_{0j}^{d}, \tag{2}$$

- and the total energy required to collect unused vehicles

$$\min \sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{d=1}^{D}(1+\sum_{v=1}^{N} prec_{vi}^{d})m_{ij}^{d}\ pow\ d_{i,j}/s, \tag{3}$$

subject to the following constraints:

- All parked vehicles are visited exactly once:

$$\sum_{d=1}^{N}\sum_{i=1}^{N} m_{ij}^{d} = 1, \forall j \in [1,N] \tag{4}$$

- No driver stays at a parked vehicle:

$$\sum_{i=0}^{N} m_{il}^{d} - \sum_{j=0}^{N} m_{lj}^{d} = 0, \forall l \in [1,N], \forall d \in [1,D] \tag{5}$$

- All drivers start at the rental station:

$$\sum_{j=1}^{N} m_{oj}^{d} = 1, \forall d \in [1, D] \tag{6}$$

- All drivers finish at the rental station:

$$\sum_{j=1}^{N} m_{jo}^{d} = 1, \forall d \in [1, D] \tag{7}$$

- No leading vehicle consumes more than its residual energy:

$$\sum_{i=0}^{N} \sum_{j=0}^{N} m_{ij}^{d} \; pow \; d_{ij}/s \leq res_{d}, \forall d \in [1, D] \tag{8}$$

- No following vehicle consumes more than its residual energy:

$$\sum_{i=0}^{N} \sum_{j=0}^{N} m_{ij}^{d} \; prec_{vi}^{d} \; pow \; d_{ij}/s \leq res_{v}, \; \forall v \in [1, N], \forall d \in [1, D] \tag{9}$$

- No subtour:

$$u_{i}^{d} - u_{j}^{d} + N m_{ij}^{d} \leq N - 1, \forall i \in [1, N], \; \forall j \neq i \in [1, N], \forall d \in [1, D], \tag{10}$$

  where u_{i}^{d} denotes the number of followers in the platoon led by the driver d when visiting location i. This subtour elimination constraint [17] imposes that $\forall d \in [1, D]$, when $m_{ij}^{d} = 0$, $u_{i}^{d} - u_{j}^{d} \leq N - 1$ and, when $m_{ij}^{d} = 1$, $u_{j}^{d} \geq u_{i}^{d} + 1$.

- Each platoon is limited to a maximum of Q vehicles:

$$\sum_{i=0}^{N} \sum_{j=1}^{N} m_{ij}^{d} < Q, \forall d \in [1, D] \tag{11}$$

- m_{ij}^{d} a binary decision variable:

$$m_{ij}^{d} \in \{0, 1\}, \forall i \in [0, N], \forall j \neq i \in [0, N], \forall d \in [1, D]. \tag{12}$$

- $prec_{ij}^{d}$ a binary variable:

$$prec_{ij}^{d} \in \{0, 1\}, \forall i \in [0, N], \forall j \in [1, N] \tag{13}$$

Solving this multi-objective optimization problem consists in finding the values of:

- the decision variable $m_{ij}^d,\ \forall i \in [0,N], \forall j \neq i \in [0,N], \forall d \in [1,D]$. By definition $m_{ii} = 0\ \forall i \in [0,N]$.
- the binary variable $prec_{ij}^d,\ \forall i \in [0,N], \forall j \neq i \in [1,N], \forall d \in [1,D]$. By definition $prec_{ii} = 0\ \forall i \in [0,N]$.

Two necessary feasibility conditions related to the residual energy of vehicles can be given. If one of these conditions is not met, the PROPAV problem is not feasible:

- *Feasibility condition for the parked vehicles:* If there exists a parked vehicle that does not have enough residual energy to return to the rental station,

$$\exists v \in [1,N] \text{ such that } d_{v,o}\ pow/s > res_v. \tag{14}$$

- *Feasibility condition for the leader vehicles:* If there exists a parked vehicle such that no leader vehicle has enough energy to pick up this vehicle and return to the rental station,

$$\begin{aligned} &\exists v \in [1,N] \text{ such that } \forall d \in [1,D], \\ &\qquad (d_{ov} + d_{v,o})\ pow/s > res_d. \end{aligned} \tag{15}$$

3.3 PROPAV as a game theory problem

In this paper, we formulate the platoon trajectory optimization problem as a coalition game. A coalition formation game is a model of game theory, where players form coalitions in order to maximize their payoff resulting from their membership of a coalition. In the PROPAV problem, the players are the parked vehicles and each coalition represents a platoon. Hence, the number of coalitions formed represents the number of platoon leaders required to collect the parked vehicles.

3.3.1 Definitions

A coalition formation game can be considered as a couple $(\mathcal{N}, f)$ where:

- $\mathcal{N}$ is a finite set of players. Let N denote the number of players. For the sake of simplicity, any player is denoted by $i \in [1,N]$.
- f is a value function that assigns a real value to each coalition C. Again, for the sake of simplicity and without loss of generality, we assume that the value of the coalition is given as a payoff to each player that is member of this coalition.

For each player $i \in [1,N]$, let $f_i(C)$ be the payoff of player i when it belongs to coalition C. According to our assumption, we have:

$$\forall i \in [1,N],\ f_i(C) = \begin{cases} f(C) & \text{if } i \in C, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

Players are assumed to be rational. Players play one after the other. We introduce the concept of *round*, such that in one round, each player plays exactly once.

3.3.2 Hedonic coalition formation

A coalition formation is said to be hedonic, if and only if the following two conditions are met [14]:

- The payoff of any player i member of coalition C depends only on the players that are members of C.
- Each player joins a coalition that improves its payoff.

Whatever the initial partition, the hedonic coalition game converges towards a Nash stable and individually stable partition [14].

PROPAV is modeled here as a hedonic coalition game with N rational players, where each player is a parked vehicle to collect. Initially, each parked vehicle constitutes a coalition. After convergence, each coalition represents the set of vehicles belonging to a same platoon driven by a human. The coalition game ensures that:

- any player belongs to exactly one coalition. Any parked vehicle is collected by exactly one human driver;
- the number of coalitions formed is the number of human drivers needed to drive the parked vehicles back to the rental station.

The value of a coalition is computed in order to:

- strongly discourage a player to rejoin a coalition that it previously joined and left.
- increase with the number of vehicles collected until a limit given by the maximum number of vehicles per platoon.
- favor coalitions that for a given number of collected vehicles minimize the tour duration.

In this coalition game, each player is selfish: it leaves a coalition to join another independently of the consequences of its move on the other players. Its only goal is to increase its payoff by applying the switching rule as follows:

> *Switching rule*: any player $i \in [1, N]$ leaves its current coalition C to join coalition $C' \in \mathcal{C}$, where $\mathcal{C}$ denotes the set of coalitions present when i is playing, if and only if $f_i(C' \cup \{i\}) > f_i(C)$.

The payoff $f_i(C)$ of a player i belonging to coalition C is computed according to Algorithm 1, where:

- $MaxTime_v$ is the maximum time that vehicle v can spend in the platoon taking into account its residual energy. Knowing the power consumed by the vehicle, the constraint on the residual energy of the vehicle can easily be expressed as a maximum duration in the platoon for this vehicle. To meet the residual energy constraint, the tour of the platoon must be such that each vehicle v spends a time less than or equal to $MaxTime_v$, otherwise the payoff of the player will be very low.

Algorithm 1 Payoff of player $i \in C$.

Run by player $i \in C$
Inputs: C, N, $History_i$, res_i, res_v $\forall v \in C$, pow, $MaxTime_v$ $\forall v \in C$
Outputs: $f_i(C)$
$MaxTime_i \leftarrow res_i/pow$
$OverLimit \leftarrow 0$
for each vehicle $v \in C \cup \{i\}$ **do**
    $NewTime_v \leftarrow$ time spent by v in an optimized tour visiting $C \cup \{i, o\}$
    if $NewTime_v > MaxTime_v$ **then**
        $OverLimit \leftarrow 1$
    end if
end for
if $C \in History_i$ or $|C| > Q$ or $OverLimit$ **then**
    $f_i(C) = -\infty$ // *discourage i to join C*
else
    if $|C| = 1$ **then**
        $f_i(C) = -MaxDistance - N - 1$ // *encourage i to join a coalition with more members if possible*
    else
        $f_i(C) = \max_{k \in C} d_{k,centroid(C)} - N/|C|$ // *encourage i to join the closest coalition if possible*
    end if
end if

- $centroid(C)$ denotes the centroid of the parked vehicles members of coalition C. A player is discouraged, by a poor payoff, from joining a coalition whose maximum distance of its members to the new centroid would increase; this distance is given by $\max_{k \in C} d_{i,centroid(C)}$.
- $|C|$ is the size of the coalition (i.e. the number of its members). This size should never exceed Q, otherwise the payoff of the player will be very low. Notice that if all coalitions had the size of coalition C, the number of coalitions would be $N/|C|$.
- $MaxDistance$ is the maximum distance between two parked vehicles. Notice that the value $-MaxDistance - N - 1$ is a payoff value that is never reached by a parked vehicle belonging to a coalition with several members. This means that any player is encouraged to join a coalition with several members rather than a partition of which it would be the only member.
- $History_i$ is the history of player i. It contains all the coalitions that player i joined. Any player is strongly discouraged from joining a coalition it already joined in the past by means of a very low payoff.

Notice that Algorithm 1 requires the computation of an optimized tour in each coalition tested. For the performance evaluation reported in Section 4, the 2-opt algorithm [18] is used for this purpose.

Algorithm 2 determines the formation of coalitions. It starts from the initial partition where each player is alone in its own coalition. Since the hedonic coalition game converges to a final stable solution whatever the initial partition, the algorithm stops when the convergence is reached. Thus, in the last round of Algorithm 2, no player can increase its payoff by leaving its current coalition to join another one. A null number of coalition switches is thus the stopping criterion of the algorithm.

4 Performance evaluation

In this section, we first present the framework used for the simulations. We then discuss the simulation results obtained by game theory and finally compare its performances to two well-known algorithms used to solve many multi-objective optimization problems: MOPSO [15] and NSGA-II [12], which are the multi-objective version of Particle Swarm Optimization and Non-dominated Sorting Genetic Algorithm, respectively. Both select the non-dominated solutions among the solutions explored to build Pareto fronts.

4.1 Simulation framework

We implemented these methods in Java and carried out the simulations on Intel Core i7 computers equipped with Windows 10. We evaluated the performance of these algorithms using 7 configurations (see Table 2) with different numbers of

Algorithm 2 Coalition formation algorithm for PROPAV.

Inputs: $\mathcal{N}$, N, *set of initial coalitions* C_0
Outputs: *set of Coalitions* C_{final}
Initialize the initial coalitions with a coalition per parked vehicle
$C_0 \leftarrow \{\{1\}, \{2\}, \ldots \{N\}\}$
$r \leftarrow 0$ // *current round number*
$C_r \leftarrow C_0$ // C_r *set of coalitions at round* r
repeat
    $change \leftarrow 0$ // *is set to 1 if* C_r *is changed in* r
    $P \leftarrow \{\varnothing\}$ // P *is the current set of players having already played in round* r
    for $k \leftarrow 1; k < N; k \leftarrow k + 1$ **do**
        Select a random player i, *while* $i \in \mathcal{N} - P$
        $P \leftarrow P \cup \{i\}$ // *add* i *to the set of already played*
        $Preferred \leftarrow Current$ // $i \in Current \in C_r$
        $Payoff \leftarrow f_i(Preferred)$
        for $j \leftarrow 0; j < |C_r|; j \leftarrow j + 1$ **do**
            Calculate $f_i'(C_j \cup \{i\})$ // *payoff of* i *if* i *joins* C_j
            if $f_i'(C_j \cup \{i\}) > Payoff$ **then**
                $Preferred \leftarrow C_j$
                $Payoff \leftarrow f_i(Preferred \cup \{i\})$
                $change \leftarrow 1$
            end if
        end for
        if $Preferred \neq Current$ **then**
            i *joins* $Preferred$
            Remove i *from* $Current$ in C_r
            Add i *in* $Preferred$ *in* C_r
            $History_i \leftarrow History_i \cup \{Preferred\}$
            if $Current == \{\varnothing\}$ **then**
                Remove $Current$ *from* C_r
            end if
        end if
    end for
    $r \leftarrow r + 1$
until $change = 0$
    return C_r // *the final set of coalitions*

Table 2: Simulation parameters.

Simulation parameter	Value
Number of parked vehicles	10, 40, 100, 150, 200, 250, 300
Geographic distribution of vehicles	uniform
Parking lot limits	defined by the four points of coordinates: • (lat: 48.8367, long: 2.1015) • (lat: 48.8369, long: 2.1026) • (lat: 48.8367, long: 2.1026) • (lat: 48.8365, long: 2.1016)
Vehicle's energy level	uniform distribution in [10, 100]

Table 3: Parameter settings for NSGA-II and MOPSO.

Algorithm	Parameter	Symbol	Value
NSGA-II	crossover probability	C_r	0.9
	mutation probability	μ	0.1
MOPSO	local coefficient	φ_1	2.0
	social coefficient	φ_1	2.0
	inertia weight	w	0.5
	population size	τ	40

vehicles {10; 40; 100; 150; 200; 250; 300}. The parameters specific to NSGA-II and MOPSO, which determine their performance, are given in Table 3.

The input of our proposed optimization solution is a JSON file that contains information about platoon leaders and a list of the automated vehicles involved. The output of the game theory algorithm is another JSON file containing the set of the platoons' trajectories with their associated leader and vehicle IDs. An example of input an output JSON files defining 2 trajectories is given in Appendix B.

Each result depicted in Figures 2 to 6 and in Table 4 is the average of 15 simulation runs. We used OSRM[5], an online API to compute the shortest paths between the different locations of the parked vehicles.

All the evaluation criteria used in this paper are quantified as a function of the number of vehicles to collect. We distinguish two types of evaluation criteria, which are related to:

- *the quality of the solution found* which concerns the three objectives to minimize, namely 1) the number of platoons, 2) the maximum tour duration, and 3) the total energy consumed.
- *the complexity of the strategy used to get the solution*, which is measured by 1) the average number of rounds needed to get the final coalitions, 2)

[5] *OSRM: Open Source Routing Machine*, `http://project-osrm.org/`

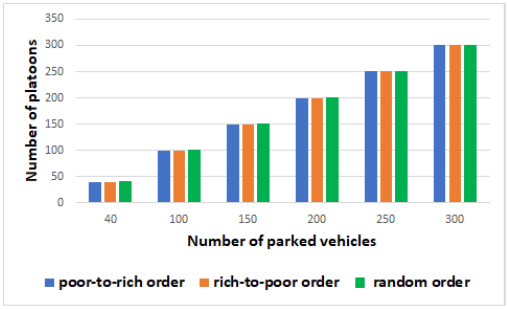

Figure 2: The average number of platoons as a function of the number of parked vehicles.

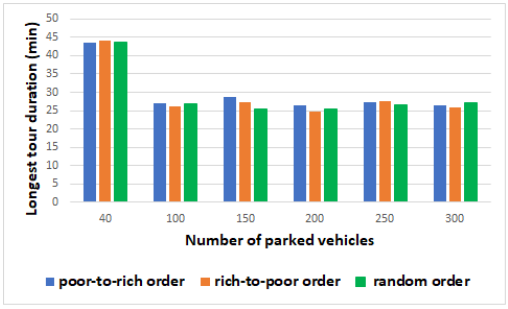

Figure 3: The average longest tour duration as a function of the number of parked vehicles.

the simulation duration, and 3) the total number of switches made by the players. This number stands for the total number of coalition changes made by the players.

4.2 Game theory results

4.2.1 Quality of the solution

We first study the impact of the number of parked vehicles to collect on 1) the number of platoons formed, 2) the longest tour duration and 3) the total energy consumed to collect all these parked vehicles.

Since the order in which the players play an iteration may influence the quality of the solution generated, we compare the results obtained with different orderings. Before starting a new iteration, the order in which the players will play this iteration is determined. Three orders are usually considered:

- *the random order*, where at each iteration, the order of players is selected randomly. This is the simplest strategy and does not require any additional computation.

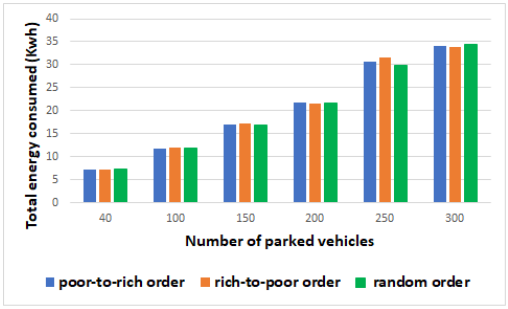

Figure 4: The average total energy consumed as a function of the number of parked vehicles.

- *the poor-to-rich order*, where before starting a new iteration, the players evaluate their payoff. The player with the smallest payoff is invited to play first. With this order, all the players play according to the increasing order of their payoff in the iteration considered.
- *the rich-to-poor order* is the reverse order: the player with the greatest payoff is invited to play first. All players play according to the decreasing order of their payoff in this iteration.

Notice that the rich-to-poor strategy and the poor-to-rich strategy require sorting the player payoffs, unlike the random strategy. The question is: does this additional computation on the players' payoffs improve the quality of the solution obtained? Simulation results are depicted in Figure 2 to 4.

As depicted in Figure 2, the average number of platoons increases with the number of parked vehicles. The play order has no impact on the number of platoons formed. The three strategies converge to the optimal number of platoons, which is equal to the number of parked vehicles divided by the maximum size of a platoon.

The longest tour duration averaged on multiple simulations is depicted in Figure 3. It starts by decreasing for a number of parked vehicles in $[40, 100]$, where the three play orders provide very similar results for the longest tour duration. For a number of parked vehicles in $[100, 300)$, the longest tour duration stabilizes at a value less than 29 s for the three play orders. However, the random order provides the smallest value of the longest tour duration. The rich-to-poor order gives better results than the poor-to-rich. This stabilization can be explained by the fact that on the one hand, the number of vehicles per platoon is limited to Q and on the other hand, the energy of each vehicle is limited.

The total energy consumed, averaged on multiple simulations, is depicted in Figure 4. As expected, it increases with the number of parked vehicles to collect. The three orders under study give almost the same results, and the random order provides the smallest total energy consumed for any number of parked

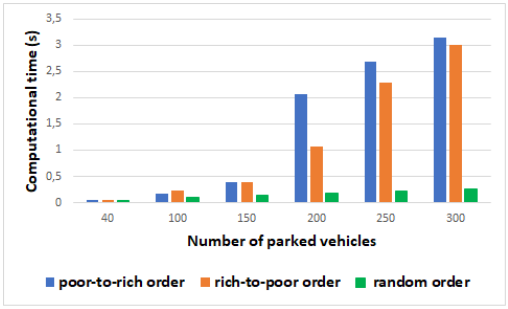

Figure 5: The average computation time as a function of the number of parked vehicles.

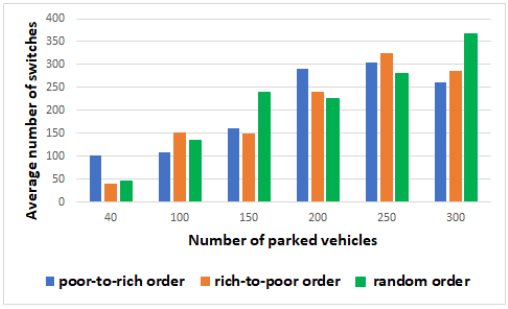

Figure 6: The average number of switches as a function of the number of parked vehicles.

vehicles less than 300.

As a conclusion, the best quality solution is provided by the random order.

4.2.2 Complexity of the strategy

The complexity of each strategy is evaluated by measuring the impact of the number of parked vehicles to collect on the computation time and on the number of coalition switches. Figure 5 depicts the average computation time as a function of the number of parked vehicles. Since the rich-to-poor order and the poor-to-rich order require sorting the players' payoffs, the computation time strongly increases with the number of parked vehicles, whereas it is worth noting that it increases very weakly with the random order.

The total number of switches averaged on multiple simulations is illustrated in Figure 6. It tends to increase with the number of parked vehicles. This can be explained by the increase in the number of coalitions formed, which leads to a greater number of coalitions improving the current payoff of the player. With regard to the six configurations tested, it can be observed that the three ordering methods produce comparable results.

As a conclusion, since the random order provides a solution of excellent quality while requiring the smallest computation time, it is the strategy used to compare the coalition game with MOPSO and NSGA-II.

4.3 Comparison with MOPSO and NSGA-II

The comparison between game theory, MOPSO and NSGA-II is made first by assuming no computation limits. The results are given in Table 4, where each time is expressed with a number of decimals corresponding to the millisecond. For each configuration evaluated, the best results appear in bold as well as the best strategy.

Table 4: Performance Comparison between Game theory and MOPSO, NSGA-II.

# vehicles	Method	# platoons	Longest tour duration (min)	Tour energy (kWh)	Computation time (s)
10	**Game theory**	**2**	**29.81**	**1.42**	**0.02**
	MOPSO	2	31.12	1.49	1.20
	NSGA-II	2	33.10	1.64	0.32
40	**Game theory**	**8**	**35.31**	**6.50**	**0.07**
	MOPSO	8	48.09	7.66	3.96
	NSGA-II	9	39.93	7.10	0.45
100	**Game theory**	**20**	**26.11**	**10.05**	**0.15**
	MOPSO	27	36.98	11.51	19.74
	NSGA-II	22	30.69	12.89	6.23
150	**Game theory**	**30**	**24.95**	**12.65**	**0.23**
	MOPSO	44	36.86	15.83	69.54
	NSGA-II	37	27.94	17.37	16.68
200	**Game theory**	**40**	**23.98**	**21.11**	**0.29**
	MOPSO	62	32.35	22.12	111.06
	NSGA-II	55	26.08	22.00	44.43
250	**Game theory**	**50**	**26.82**	**28.07**	**0.40**
	MOPSO	67	43.56	29.35	195.54
	NSGA-II	63	31.26	34.42	84.21
300	**Game theory**	**60**	**24.48**	**32.26**	**0.47**
	MOPSO	72	30.87	33.83	560.28
	NSGA-II	68	27.95	35.11	314.41

Game theory strongly outperforms both MOPSO and NSGA-II in terms of the quality of the solution produced. It always provides the smallest number of platoons, which is very important in practice, since it represents a great benefit for the shared transportation operator. Furthermore, game theory provides the smallest longest tour duration and the lowest amount of energy consumed, which

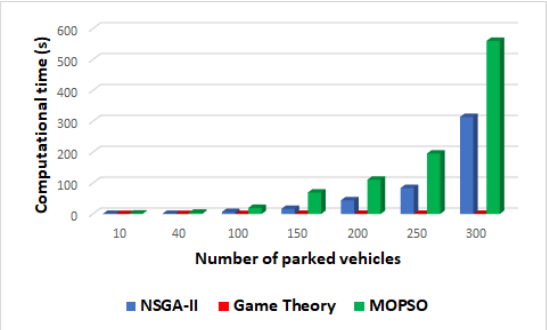

Figure 7: The average computation time as a function of the number of parked vehicles for game theory, MOPSO and NSGA-II

are also sources of benefit. For instance, for a number of parked vehicles equal to 100, game theory gives 1) a number of platoons 35% smaller than MOPSO and 10% less than NSGA-II, 2) a longest tour duration 42% smaller than MOPSO and 18% smaller than NSGA-II, and 3) a total amount of energy consumed 14% less than MOPSO and 28% less than NSGA-II. For a number of parked vehicles equal to 300, game theory gives 1) a number of platoons 20% smaller than MOPSO and 13% less than NSGA-II, 2) a longest tour duration 26% smaller than MOPSO and 14% smaller than NSGA-II, and 3) a total amount of energy consumed 5% less than MOPSO and 9% less than NSGA-II.

With regard to the number of platoons and the longest tour duration, NSGA-II outperforms MOPSO except for 10 and 40 parked vehicles, whereas MOPSO outperforms NSGA-II for the energy consumed except for 40 parked vehicles.

With regard to the computation time required to produce the solution, game theory always requires the shortest computation time (Figure 7): it always converges in less than a second, whereas MOPSO (resp. NSGA-II) can require up to 9'20" (resp. 5'14"). This difference is very important in an operational setting, where a solution could become obsolete if it takes too long to compute it.

Finally, the evolution of the quality of the solution provided by game theory, MOPSO and NSGA-II is compared as a function of computation time. Such simulation results can be used to decide whether it is worth waiting longer to obtain a better solution. We consider a number of parked vehicles equal to 300.

We focus on the first optimization criterion, which is the number of platoons. This is the most important criterion from the economic point-of-view, since it can lead to a reduction in operating costs. Simulation results are depicted in Figure 8. Game theory starts with a number of platoons equal to the number of parked vehicles and very quickly converges to the optimal number of platoons, which is 60. Both NSGA-II and MOPSO start with 100 platoons. After 13 s, NSGA-II provides a smaller number of platoons than MOPSO. The difference seems to remain constant, even if both algorithms continue to very slowly decrease the

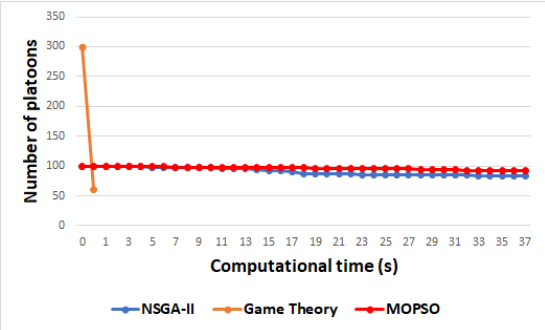

Figure 8: Evolution of the number of platoons over computation time for game theory, MOPSO and NSGA-II with 300 parked vehicles.

number of platoons. Thus, for game theory, it is better to wait until the end of the program, which is very fast (i.e. 0.468 s). For NSGA-II and MOPSO, the decrease in the number of platoons is extremely slow, with an advantage to NSGA-II that always gives a number of platoons less than MOPSO.

As a conclusion, game theory is by far the most efficient strategy.

5 Discussion

With regard to the excellent quality of the solution produced by the coalition game in a very short time, the coalition game with a random order of players very greatly outperforms both MOPSO and NSGA-II. Two questions arise from this: 1) which reasons can explain this result? and 2) can the coalition game be applied to problems other than PROPAV?

The most obvious reason for the low complexity of the coalition game is the simplicity of its algorithm: each player tries to maximize its payoff by joining the coalition that gives the highest payoff. The key to a successful coalition game lies in the design of the payoff function.

The payoff function is designed in order to meet the following two objectives:

- First, it should discourage players from forming unacceptable coalitions. A coalition is said to be unacceptable if and only if it violates the constraints of the problem to solve. For example, in the PROPAV problem, any coalition with more than Q members or any coalition which requires an amount of energy higher than the residual energy of one of its members. Consequently, the payoff of any unacceptable coalition is set to the worst value. Furthermore, the coalition game presented in this paper requires that any member should never join a coalition that it had already joined and left in the past. This is in order to ensure convergence and it is a necessary condition whatever the problem considered.

- Second, it should give an incentive to players to join a better coalition. A coalition is said to be better if and only if it optimizes at least one optimization criterion without degrading the other optimization criteria. Hence, the payoff function should reflect the improvement of the coalition with regard to the optimization criteria. Thus, it will make switching to a better coalition a greater incentive for the players. Usually, in the initial state of the coalition game presented in this paper, each player forms its own coalition. In the PROPAV problem, the payoff function of any coalition C meeting all the constraints is set to $\max_{k \in C} d_{k,centroid(C)} - N/|C|$, where $d_{k,centroid(C)}$ is the Euclidian distance of k to the centroid of C, N is the number of players and $|C|$ the size of C. This payoff encourages players to join nearby coalitions, as doing so minimizes the tour duration and the energy consumed.

As a conclusion, the coalition game can be applied to any clustering problem where the clusters are subject to constraints and certain criteria need to be optimized. All the subtlety required to apply the coalition game lies in the design of the payoff function, which should reject unacceptable clusters and favor clusters optimizing the optimization criteria.

6 Conclusion

In this paper, we defined the PROPAV problem as a coalition game, where the players are the electric automated vehicles to be picked up and returned to the rental station. The three optimization criteria are 1) the number of platoons, which is minimized, 2) the tour duration of each platoon leader, which is minimized and 3) the total energy consumed is also minimized. Multiple constraints are taken into account such as the maximum number of vehicles per platoon, and the residual energy of each vehicle in the platoon. The coalition game very quickly converges to a set of coalitions, where each coalition is a platoon driven by a platoon leader.

Simulation results obtained for various configurations where the number of vehicles to pick up ranges from 10 to 300 show that game theory always provides the best quality solution in terms of the three optimization criteria. The coalition game always provides the optimal number of platoons, which results in 20% fewer platoons than MOPSO and 13% fewer platoons than NSGA-II in the case of 300 vehicles to collect.

Furthermore, the complexity of the coalition game evaluated by both the computation time and the number of switches is much smaller than that of MOPSO and NSGA-II. The computation time remains below 1 s for all tested cases, whereas the other methods require several minutes. This difference is crucial in an operational setting.

To build upon this study, three further aspects could be taken into consideration in future work. First, we can extend the optimization approach based on this coalition game so as to take into account multiple rental stations. One strategy would be to assign an additional unknown to each unused vehicle, that is

the rental station to which it should be brought. Alternatively, only the number of vehicles to return to each rental station can be specified. This second solution is probably the most suited for a carsharing system. However, it would only make sense when coupled with an algorithm that decides what stations should be refilled to match demand.

Second, additional objectives or constraints can be considered in the PROPAV problem to better reflect real-world conditions. For example, signalized intersections can cause platoon dispersion and the separation of some vehicles from the platoon to which they belong. Hence, the number of road crossings in the platoons' tour should be kept as low as possible [8].

Finally, in the long term, all the problems that are well suited to this coalition game should be characterized.

References

[1] World Health Organization et al., *Ambient air pollution: A global assessment of exposure and burden of disease*, published by World Health Organization, 2016.

[2] B. Zhou and K. M. Kockelman, *Opportunities for and Impacts of Carsharing: A Survey of the Austin, Texas Market*, International Journal of Sustainable Transportation, vol. 5, no. 3, pp. 135–152, 2011.

[3] M. Nourinejad and M. J. Roorda, *Carsharing operations policies: a comparison between one-way and two-way systems*, Transportation, vol. 42, no. 3, pp. 497–518, 2015.

[4] M. Dror, D. Fortin and C. Roucairol, *Redistribution of self-service electric cars: A case of pickup and delivery*, Technical Report W.P. 3543, INRIA-Rocquencourt, France, 1998.

[5] P. Daviet and M. Parent, *Platooning techniques for empty vehicle distribution in the Praxitèle project*, Proceedings of the 4th IEEE Mediterranean Symposium on New Directions in Control & Automation, Maleme, Greece, June 1996.

[6] M. Marouf, E. Pollard and F. Nashashibi, *Automatic parallel parking and platooning to redistribute electric vehicles in a car-sharing application*, IEEE Intelligent Vehicles Symposium IV, Dearborn, Michigan, United States, Jun 2014, pp. 486–491.

[7] C. Flores, P. Merdrignac, R. de Charette, F. Navas, V. Milanés, and F. Nashashibi, *A Cooperative Car-Following/Emergency Braking System With Prediction-Based Pedestrian Avoidance Capabilities*, IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 5, pp. 1837–1846, 2019.

[8] M. Hadded, J-M. Lasgouttes, F. Nashashibi, I. Xydias. *Platoon Route Optimization for Picking up Automated Vehicles in an Urban Network*, ITSC

2018, 21st IEEE International Conference on Intelligent Transportation Systems, Nov 2018, Maui, United States. https://hal.inria.fr/hal-01880388

[9] N. Hafez, M. Parent and J. M. Proth, *Management of a pool of self-service cars*, IEEE Intelligent Transportation Systems ITSC, Oakland, USA, Aug 2001, pp. 943–948.

[10] A. Febbraro, N. Sacco, M. Saeednia, *One-way carsharing: Solving the relocation problem*, Transportation Research Board 91st Annual Meeting, 2012.

[11] M. Bruglieri, A. Colorni and A. Lue, *The vehicle relocation problem for the one-way electric vehicle sharing, an application to the Milan Networks*, Social and Behavioral Sciences, vol. 64, no. 4, pp. 292–305, 2014.

[12] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, *A fast and elitist multi-objective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002.

[13] B. M. Baker and M. Ayechew, *A genetic algorithm for the vehicle routing problem*, Computers & Operations Research, vol. 30, no. 5, pp. 787–800, 2003.

[14] W. Saad, H. Zu, T. Basar, M. Debbah, A. Hjrungnesng, *Hedonic coalition formation for distributed task allocation among wireless agents*, IEEE Trans. on Mobile Computing, vol. 10, no. 9, Sep. 2011.

[15] Y. Marinakis, M. Marinaki and A. Migdalas, *Particle Swarm Optimization for the Vehicle Routing Problem: A Survey and a Comparative Analysis*, Handbook of Heuristics, pp. 1–34, Oct. 2017.

[16] N. Ben Hassine, P. Minet, M.-A. Koulali, M. Erradi, D. Marinca, D. Barth, *Coalition Game for Video Content Clustering in Content Delivery Networks*, the 14th Annual IEEE Consumer Communications & Networking Conference, CCNC 2017, Las Vegas, Nevada, January 2017.

[17] C. E. Miller, A. W. Tucker and R. A. Zemlin, *Integer Programming Formulation of Traveling Salesman Problems* Journal of the ACM, Vol. 7 I. 4, pp. 326–329, 1960.

[18] G. A. Croes, *A method for solving traveling-salesman problems*, Operations Research, vol. 6, no. 6, pp. 791–812, 1958.

A Example of a JSON input file

The JSON input file given below defines the geographical location of the rental station and the limits of the area where vehicles are parked, the allocated computation time, and vehicle database with 2 automated vehicles identified by `"type": "automated"` and 1 platoon leader identified by `"type": "leader"`.

Each vehicle has its own charge level. In the example, one automated vehicle is a Volkswagen Beetle, and the other is a Renault Clio.

```
{"parkings": /* parking lots location */ [
  {
    "id": "PA12",
    "latitude": 48.836995,
    "longitude": 2.103342,
    "limits": [
        {
            "coordinates": {
                "latitude": 48.83673885,
                "longitude": 2.1015541
            }
        },
        {
            "coordinates": {
                "latitude": 48.8369617,
                "longitude": 2.10262325
            }
        },
        {
            "coordinates": {
                "latitude": 48.83676733,
                "longitude": 2.10269994
            }
        },
        {
            "coordinates": {
                "latitude": 48.8365594,
                "longitude": 2.10163369
            }
        }
    }
  ],
  "vehicles": /* vehicles database */  [
  {
        {
        "chargeLevel": 20,
        "currentStatus": "waiting mission",
        "id": "Leader1",
        "lastConnection": "14:12:10",
        "latitude": 48.8688815,
        "longitude": 2.3920687,
        "signalLevel": 20,
        "type": "leader"
```

```
    },
    {
        "chargeLevel": 55,
        "currentStatus": "waiting mission",
        "id": "volkswagen_coccinelle_1",
        "lastConnection": "11:00:00",
        "latitude": 48.8590063052678,
        "longitude": 2.36838227211365,
        "signalLevel": 18,
        "type": "automated"
    },
    {
        "chargeLevel": 72,
        "currentStatus": "waiting mission",
        "id": "renault_clio_1",
        "lastConnection": "11:00:00",
        "latitude": 48.8616624758818,
        "longitude": 2.38818416296454,
        "signalLevel": 98,
        "type": "automated"
    }
],
"allocatedComputeTime": 60 /* in seconds */
}
```

B Example of a JSON output file

The JSON output file given below defines two platoons, with for each: 1) its platoon leader, 2) the id of all the platoon members and 3) its trajectory.

```
["platoons":
  {
    "platoon_leader_id": "Leader1",/* The ID of
       platoon leader */
    "vehicles": [
        {
            "cid":"citroen_E-mehari_4"
        },
        {
           "cid":"renault_clio_1"
        },
         {
           "cid":"peugeot_508_2"
        }
```

```
            ] /* the list of parked vehicles that will
                 be retrieved by the platoon leader */
    "platoon_trajectory" : /* The path with its
       associated way-points that will be followed by
       the platoon leader */
        {
                .
                .
        }
}

  {
    "platoon_leader_id" : "Leader2",
    "vehicles": [
        {
            "cid":"bmw_serie3_2"
        },
        {
           "cid":"peugeot_2008_1"
        },
        {
           "cid":"peugeot_5008_1"
        }
            ]
    "platoon_trajectory" :
        {
                .
                .
        }
    }
  }
]
```