A New Parking Space Allocation System based on a Distributed Constraint Optimization Approach

Atik Ali[1], Souhila Arib[2] and Samir Aknine[3]

[1] *Research Center on Scientific and Technical Information, CERIST, Higher National School of Computer Science, ESI ex INI, Algeria*

[2] *CY Tech Cergy Paris Université, 33 Boulevard du port 95000 Cergy, France*

[3] *Laboratoire LIRIS, Université Claude Bernard Lyon 1, UCBL 69622 Villeurbanne, France*

Keywords: Parking Allocation, DCOPs, Max-Sum Algorithm.

Abstract: This paper develops and evaluates a new decentralized mechanism for the allocation of parking slots in downtown, using a distributed constraints optimization approach (DCOP). Our mechanism works with the multi-parking/multi-zone model, where vehicles are connected and can exchange information with the distributed allocation system. This mechanism can reach the minimal allocation costs where vehicles are assigned to the parking lots with the best possible aggregated user costs. The cost is calculated based on driver's aggregated preferences over slots. We empirically evaluated the performance of our approach with randomly generated costs and tested on three different configurations. The evaluation shows the performance of each configuration in terms of runtime and volume of exchanged data.

1 INTRODUCTION

Parking demand is determined by the attractiveness of the destination, with city centers being the most attractive areas. These areas are experiencing a growing shortage of parking places, and drivers generally need to spend a significant amount of time circling the blocks around their destination searching and waiting for available parking spaces. In this problem, the difficulty lies in the fact that the requirements and the destinations of the drivers vary from one driver to another. Many works have been carried on to deal with this last issue. For instance, (Zeenat et al., 2018) proposed a multiple criteria based parking space reservation algorithm that can be used to reserve a space for users, and to deal with their requirements in a fair way by using the normalized weighting of each criteria score for all options relating to each user's profile. In (Boudali and Ouada, 2017), the authors proposed a system that handles user's preferences by ensuring an online space allocation based on real-time information and by optimizing driver's preferences with respect to a set of operational constraints, such as, the bound on parking fees, bounds on distances, time interval that have to be satisfied for each reservation, etc. In (Ayala et al., 2011), the authors proposed to allocate parking slots via negotiations. This work uses a game theoretic approach to allocate the suitable place for the suitable vehicle, or the shortest way to the suitable spot via a routing algorithms (Hedderich et al., 2017), or via smart reservation system as in (Kazi et al., 2018). All these works, each in its own way, claim to reduce the traffic in urban areas, and to alleviate its negative related effects, from the reduction of air pollution, the reduction of fuel consumption, to the reduction of cursing time, social anxiety, etc. Besides, with the advent of IoT technology, the proliferation of smart-phones, and on-board navigation systems, it is already possible for a user to be linked in a real-time with parking space allocation applications that lead their users to the best possible spot to park. In this paper, we study the setting up of a parking management system to respond without a centralized control to parking requests in downtown parking areas, with the aim of limiting both the time of response and the cost of communication. Indeed, commercial fleet management solutions that require real-time data collections and exploitation, especially in the Cloud like parking allocation, carpooling, become rapidly costly depending on the rate of the data collections and the volume of data to be processed (Picard et al., 2018). We thus focus on finding efficient solutions for the allocation of parking slots. To that end, we formulate the problem as a distributed

Ali, A., Arib, S. and Aknine, S.
A New Parking Space Allocation System based on a Distributed Constraint Optimization Approach.
DOI: 10.5220/0010249201960204
In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence (ICAART 2021) - Volume 2*, pages 196-204
ISBN: 978-989-758-484-8

constraint optimization problem which has shown to be effective in vehicle routing optimization (Léauté et al., 2010), auctions and resource allocation problems (Weiss et al., 2017). The decentralized approach has the benefit of distributing the main computations across all available devices, thus giving more computation power to the system. Here we do not claim to propose a solution that is faster than the centralized models but rather a solution that requires less volume of data to be processed than the centralized ones[1]. Our main contributions are:

1. We propose a new distributed parking allocation model that we test in three architectural shapes for a set of n connected vehicles.
2. We choose the best DCOP solver, and show how to extend the model from the static case to the dynamic case.

This article is organized as follows: section 2 presents the related work. Section 3 presents DCOPs in general, then our problem formulation and solution in subsections 3.1, 3.2 and 3.3. An illustrative example is given in subsection 3.4, whereas subsections 3.5, 3.6 and 3.7 are devoted to the presentation of our three approaches. Section 4 shows experimentation we carried on our three approaches. Finally, the conclusion is presented in section 5.

2 RELATED WORK

Among distributed approaches, we can mention physically distributed multi-agent systems. As an example of the latter we can cite (Chou et al., 2008) in which the authors proposed an agent-based intelligent parking negotiation and guidance system which acts as a bargaining platform between parks and drivers. The platform facilitates the search for available parking spaces, the dynamic negotiation of parking fees, the reservation of parking spaces, and the derivation of optimal paths from the current location to the intended car park as well as from the car park to the final destination. This approach presents some shortcomings due to the increasing number of exchanged messages during the negotiation process as well as the possibility of non-convergence of the negotiation toward a viable solution. Distributed systems could be self-coordinated car parking methods like in (Bessghaier et al., 2012), in which the authors propose a decentralized multi-agent paradigm with one of the objectives being the coordination of distributed entities based on inter-vehicular communication (V2V) which allows vehicles to receive and broadcast information on space availability to the other vehicles of the same community. The system works without prior information on the parking spots and without central storage of information. In fact, vehicles of the same district update their local knowledge by exchanging messages which include the list of occupied and free spots at each parking and each discharge. In these categories, the vehicle has a trade-off between cooperating to collect information or competing to find a suitable parking slot. Indeed, to improve the cooperation between vehicles, a recent work (Chou et al., 2018), proposed a decentralized car parking approach for vast car park areas. Their approach called Copark is based on cooperation among vehicles through vehicle-to-vehicle communication. Their system uses a cooperation algorithm which optimizes the level of satisfaction in terms of individual and social benefits in large car parking areas. In our work, we propose a new decentralized multi-agent paradigm with distributed optimizer and using VANETs network and investigate the impact of our model on communication costs.

3 PROBLEM FORMULATION

Distributed constraint optimization problem is used in decentralized approaches. It gives more computational power to the system by distributing the computational load over multiple computation nodes. DCOP gives better scalability and adaptability, since the failure/maintenance in any node doesn't affect the whole system. Finally, it gives more power to the system by allowing it to operate in highly dynamic environment such as car park environment. A DCOP is a tuple $\{A, X, D, C, \mu\}$, where: $A = \{a_1, a_2, ..., a_{|A|}\}$ is a set of agents; $X = \{x_1, x_2, .., x_n\}$ is a set of decision variables, each decision variable x_i being controlled by an agent $a(x_i)$; and the domain $D = \{d_1, d_2, ..., d_n\}$ is a set of finite discrete domains for the decision variables such that x_i takes values in the sub-domain d_i. $C = \{c_1, ..., c_\theta\}$ is a set of constraints that assign costs where $c_i \in \mathbf{R}$. Each c_i is assumed to be known to all involved agents. The μ defines a mapping function between agents and variables from X. A solution to the DCOP is an assignment to all variables that minimizes the sum $\sum_{i=1}^{\theta} c_i$.

In our model, we consider car parks disseminated in a random manner in a given area portioned into several zones. We assume that each locality (zone) contains from 0 to λ car parks. Each single car park

[1]For technical reasons, and with respect to the context, we only focus on optimizing the calculation time, and minimizing the data volume to be processed without physically using a real distributed hardware.

belongs to a single locality (zone) z_i from the set $Z = \{z_0, z_1, z_2, \ldots, z_{|Z|}\}$. In what follows, we present some notations relative to the vehicles (or users) and to the car parks. In our model, we consider N users requesting parking in this area. We define V as the set of vehicles (or users) who are searching for a place to park: $V = \{v_x, x \mid x = \{1,2,3,...,N\}\}$. We define P as the set of car parks in our downtown, $P = \{p_k, k \mid k = \{1,2,3,...,|P|\}, \quad |P| \leq \lambda$. Let us denote $Dest_x$ as the destination of the vehicle v_x, and let us define D_{max_x} as the maximum distance allowed by the vehicle v_x from the elected car park space to the vehicle's destination $Dest_x$. We define the parameter $q(p_k)$ as the capacity of the parking number k. We define zv_x/zp_k as the zone number where is located the destination of the vehicle v_x respectively the car park p_k. We define the set of the candidate car parks $C_D(x)$ for the user x or the vehicle v_x who is seeking to park in a given zone $z \in Z$. The $c(p_k, v_x)$ is the cost when the vehicle v_x parks in that car park p_k. The $c(p_k, v_x) \neq \infty$ means that the car park p_k has a proposal to our user.

We define the set $C_D(x)_{\neq z}$ to design the car parks which do not belong to the user's zone destination but are near to the user's destination (i.e. the tolerable walking distance of user x, D_{max_x} is respected and $C_D(x)_{\neq z} \subseteq P$).

$$C_D(x) = \{p_k \mid p_k \in P, walking(p_k, v_x) \leq D_{max_x} \wedge c(p_k, v_x) \neq \infty \wedge p_k \in zv_x\} \quad (1)$$

$$C_D(x)_{\neq z} = \{p_k \mid p_k \in P \wedge p_k \notin C_D(x), walking(p_k, v_x) \leq D_{max_x} \wedge \ zv_x \neq zp_k\} \quad (2)$$

Where $walking(p_k, v_x)$ returns the time that the user needs to reach the parking. We define toc_x as the approximate occupancy time claimed by the user x and we define t_x^s and t_x^e as the start and end time interval relating to the user x. $t_x^s, t_x^e \in [1, \tau]$ where $\tau = \{1,2...,T\}$, designates the iteration current time.

$$toc_x = t_x^e - st_x^s + 1 \quad (3)$$

We define $\tau_{k.x}$ the estimated walking time between the vehicle x current car park slot and the user destination $Dest_x$. We define $Est_{x.k}$ as the average estimated travel time between the vehicle x current position and the car park of number k. We define the $Pdur_{x.k}$ as a calibration quantity which includes the stay, the traveling and the walking time parameters via the following formula:

$$Pdur_{x.k} = toc_x + Est_{x.k} + 2\tau_{k.x} - 1 \quad (4)$$

We define the subset $X_{\pi.k} \subset X$ as the allocation variables for the car park p_k where:

$$X_{\pi.k} = \{x_{i.k}, .., x_{a.k}, .., x_{n.k}, .., x_{s.k}..\} \quad (5)$$

$x_{i.k}$ designates an allocation variable for an ordinary user.

$x_{a.k}$ is the allocation variable for the elderly.

$x_{n.k}$ is the allocation variable for the commuter.

$x_{s.k}$ is the allocation variable for the sick users.

We define our allocation function $X_{x.k}$ as follows[2]:

$$X_{x.k} = \begin{cases} 1 & \text{if vehicle x is assigned at car park k.} \\ 0 & \text{otherwise.} \end{cases}$$

For a given allocation $X_{x.k}$, we associate a cost function $C_{x.k}$, where each allocation must be able to derive the cost from the parking. For any user x, the $c_{x.k}$ represents the cost of a certain user x if he chooses to park in a car park k.

3.1 Objective

Our objective is to propose to each vehicle of our set of vehicles a place to park, (i.e. to propose a set of identifiers of car parks where each vehicle can park), This set of car parks is called a configuration. Let t be the current time step and V the set of all vehicles seeking to park in the system. A configuration is a set $\phi = \{p_1, p_2, .., p_N\}$ where each p_i is the car park number assigned to each $v_i \in V$. We aim to build, for each time step t, a configuration ϕ_t for all vehicles in V that minimises their total cost of allocation. The input is the set of vehicles V_t presented in the system at the current time step and the configuration at the last time step ϕ_{t-1}. Let c_x be the allocation cost of vehicle v_x (this cost is influenced by every vehicle's priority, location and destination). Let Φ be the set of all possible configurations. Our goal is to search for a minimisation of the function f:

$$f : (t, V_t, \phi_{t-1}) \mapsto Arg \min_{\phi_t \in \Phi} \sum_{v_i \in V_t} c_i \quad (6)$$

3.2 Structural Constraints

Structural constraints are bunch of hard constraints that represent the compliance of the resource with the basic spatial allocation rules which must be respected together in order to satisfy the consumers. These structural constraints are as follow:

3.2.1 The Single Slot Constraint

The constraint (7) ensures that, at most only one vehicle is allocated to a single car park slot.

$$O(X) = \sum_{p_k \in P} X_{x.k} \leq 1, \forall v_x \in V \quad (7)$$

[2]For ease of writing $x \in \{i, a, n, s\}$.

It enforces that v_x must be assigned to at most one car park.

3.2.2 Car Park Capacity Constraint

This constraint ensures that the sum of all allocations for a given parking area never exceeds its capacity.

$$C(X) = \sum_{v_x \in V} X_{x.k} \leq q(p_k), \forall p_k \in P \tag{8}$$

3.3 The Soft Constraints

These constraints serve to favor the users on the basis of their profile, these may be violated but their violation is monetized by a given cost.

3.3.1 The Priority Constraint

For that, we define in (9) the binary relation $Typc(x_{i.k}, x_{x.k})$ of tuples of elements of $X_{\pi.k}$ where the $x_{i.k}$, as previously explained in (5), is a variable allocation for an ordinary user. $c_{i.k} - c_{x.k}$ is a cost difference between the user i and the user $x_{x.k} \in X_{\pi.k} \backslash x_{i.k}$ pertaining to the car park k. The triple equality $x_{i.k} = x_{x.k} = 1$ means that for every car park p_k in the system there are four levels of priority, one for the ordinary users, another one for the sick and the elderly persons and finally another one for the commuters.

$$Typc(x_{i.k}, x_{x.k}) = \{(x_{i.k}, x_{x.k}) \mid x_{i.k} \in X_{\pi.k}, \\ x_{x.k} \in X_{\pi.k} \backslash x_{i.k} \ , \ x_{i.k} = x_{x.k} = 1 \wedge \ c_{i.k} - c_{x.k} > 0\} \tag{9}$$

We define the binary priority constraint $\Delta(x, x')$ as follows:

$$\Delta(x_{i.k}, x_{x.k}) = \begin{cases} 0 & if \ (x_{i.k}, x_{x.k}) \in Typc(x_{i.k}, x_{x.k}) \\ 1 & otherwise. \end{cases} \tag{10}$$

3.3.2 The Travel Time Constraint

The binary relation (11) is between two users heading to a given car park k, this relation assigns a lower cost to the vehicle having estimated travel time between its current position and the car park in question.

$$Tr(x_{x.k}, x_{x'.k}) = \{ \ (x_{x.k}, \ x_{x'.k}) \ \in X^2_{\pi.k}, \\ x_k = x_{x.k} = 1 \wedge Est_{x.k} > Est_{x'.k} \\ \wedge \ c_{x'.k} - c_{x.k} > 0\} \tag{11}$$

then the equation (12) defines the travel time constraint as follows:

$$\Lambda(x_{x.k}, x_{x'.k}) = \begin{cases} 0 & if \quad (x_{x.k}, x_{x'.k}) \in Tr(x_{x.k}, x_{x'.k}) \\ 1 & otherwise. \end{cases} \tag{12}$$

3.3.3 The Parking Duration Constraint

The relation (13) is another soft binary relation that gives a lower cost to the user who is claiming to stay no longer than his competitor (again if both users are heading to the same car park) this constraint is to encourage the users to minimize their parking time.

$$E(x_{x.k}, x_{x'.k}) = \{ \ (x_{x.k}, x_{x'.k}) \ \in X^2_{\pi.k}, \\ x_{x'.k} = x_{x.k} = 1 \wedge Pdur_{x.k} < Pdur_{x'.k} \\ \wedge \ c_{x.k} - c_{x'.k} > 0\} \tag{13}$$

The parking duration constraint will be the equation (14):

$$\rho(x_{x.k}, x_{x'.k}) = \begin{cases} 0 & if \quad (x_{x.k}, x_{x'.k}) \in E(x_{x.k}, x_{x'.k}) \\ 1 & otherwise. \end{cases} \tag{14}$$

We define our allocation objective function as follows:

$$Arg \min_X f(X) = \sum_{x=1}^{N} \sum_{k=1}^{\lambda} (C_{x.k} X_{x.k}) \tag{15}$$

subject to: (7), (8), (10), (12) and (14)

We let the binary relation X from a set $V \times P$ to a set G, where G is a subset of:
$V \times P, \ X : V \times P \ \rightarrow G \subseteq V \times P$

An allocation X is optimal if it minimizes the objective function $f(X)$ and respects all the hard constraints (7) and (8) and the remaining soft constraints (10), (12) and (14) listed above. Now we define our DCOP as the tuple $\{A, X, D, C, \mu\}$, where the set A is a finite set of m agents that manage the zones and the users, $A = \{a_0, a_1, .., a_m\}$. This means that certain agents are assigned for the control of the different users or car parks or zones of the area. We define the set D as a set of finite discrete domains: $D = \{d_0, d_1, .., d_\varepsilon\}$ and the set $X = \{X_{\pi.1}, X_{\pi.2}, .., X_{\pi.\lambda}\} \cup \{n_1, n_2, .., n_\delta\}$ as the finite set of variables owned by all the agents $a_i \in A$. The subset of variables $\{X_{\pi.1}, X_{\pi.2}, .., X_{\pi.\lambda}\}$[3] takes values from the domain $[0,1]$; the variables n_δ take values from another set of discrete domains: $d_x \subset D$ where $d_x = C_D(x)_{\neq z} \cup C_D(x)$. Here the set D is nothing else than the whole car parks subsets of the system plus the two element set $\{0,1\}$. We define the set of constraints $C = \{c_0, c_1, .., c_\theta\}$ as the set of constraints over the variables of X where each constraint $c_\theta \in C$ defines a cost $\in \mathbf{R} \cup \{\infty\}$ and for each possible allocation we associate a cost corresponding to it. A solution to our DCOP consists in the optimization of the equation (15) subject to the set of constraints. Finally, μ the last element of our tuple's DCOP problem, is a

[3] See equation (5)

mapping function that assigns for every variable from X an agent $a_i \in A$ so that no variable from X is assigned to more than one agent and $\mu : X \rightarrow A$. The mapping function μ will be explained in more detail in the next sub-sections 3.5, 3.6 and 3.7.

In our formulation, we assume that the agents have the following capabilities: (1) Agents are capable of communicating with each other via the network in a cooperative manner with respect to a communication radius. (2) Users are capable to communicate with the bargaining agents which are in charge of negotiating slots via a central server. (3) Requests are sent via the communication infrastructure which can be global (e.g. cellular network, embedded system) and sent to a central server.

3.4 Illustration of Our DCOP Model with an Example

In this example, we are assuming that each car is monitored by an agent, and each agent holds four parameters: the car's destination, the car's user type, the user walking distance threshold, and the user's estimation of his occupation time. We consider a global area divided into 2 zones (central, peripheral), with 3 car parks in the central zone and one car park in the peripheral zone (illustrated by the red squared houses (See Fig. 1). We consider 6 users requesting to park, three cars requesting to park in the central zone and three cars requesting to park at the peripheral zone. Every car agent calculates the cost of the candidate car park of its own user: for instance the cost calculated via the agent of the car number 2 if he chooses to park at car park number 1 is 71, the agent of the user number 4 knows that there is no place in the same car park ($cost = \infty$). The agent of the user number 5 calculated the cost 44 with reference to car park number 4 (see Table 1)[4]. For simplicity, the set of users $y = \{1,2,3\}$ requesting to park in the central zone will have the same set of candidate car parks $C_D(y) = \{C\_park_1, C\_park_2, C\_park_3\}$(i.e. their respective destinations are inside the central area zone as shown in the Fig. 1[5]. The *user*$\{4\}$ has the set $C_D(4) = \{C\_park_4\}$ of candidate car parks and $C_D(4) = \{C\_park_2, C\_park_3\}$ as illustrated in Fig. 1. For the remaining users $\{5,6\}$, the set of the candidate car parks is the singleton $\{C\_park_4\}$ (see Table 2). Looking at the constraint graph (Fig. 2),

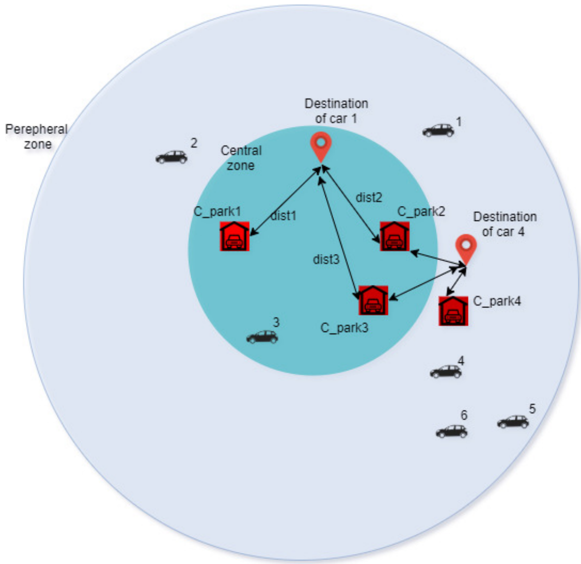

Figure 1: The red squares represent the car parks. The dist1, dist2 and dist3 represent the three distances between the car number 1 destination and the three car parks (1, 2 and 3). Circles are zones delimiters illustrating the division of the car park area into sectors based on the distance from downtown center.

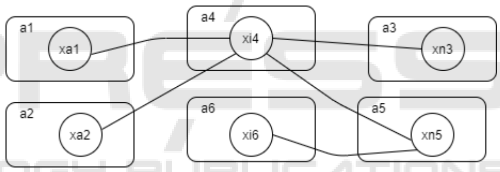

Figure 2: Constraint graph of our example with the vehicle based approach.

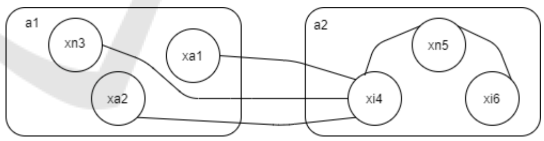

Figure 3: Constraint graph of our example with the zone based approach.

we can see that each single agent a_i is responsible of exactly one allocation variable. The figure 3 illustrates the constraint graph of the same problem with this time every zone is monitored by an agent, here we have two agents responsible for two zones. The first agent a_1 is responsible of the central zone, subsequently in charge of the variables $\{x_{a.1}, x_{a.2}, x_{n.3}\}$ representing respectively the allocation variables for the users $\{1,2,3\}$. The agent a_1 is responsible for the satisfaction of the set of constraints $C = \{(7),(8),(10),(12),(14)\}$. Similarly, agent a_2 is responsible for the peripheral zone and is thus in charge of the variables $\{x_{n.5}, x_{i.6}, x_{i.4}\}$ respectively

[4] Each cost cell is calculated on the basis of the cost accumulations of the constraints (7), (8), (10), (12), (14).

[5] This simplification is made to help explain the model. In a complete example, every single user will have a set of candidate car parks which could be calculated according to the distance D_{max_x} relating to each user, see equation (1).

pertaining to users $\{4,5,6\}$. Similarly agent a_2 is responsible for the satisfaction of the set of constraints $C = \{(7),(8),(10),(12),(14)\}$. We wrote the problem formulation in XML format, and launched the resolver using the Max-Sum algorithm from the frodo platform (Léauté et al., 2009), applying the equation (15) and with the respect of the set of constraints C cited above which generated the cost grid (see Table 1). $Arg\min_X f(X) = \sum_{x=1}^{6}\sum_{k=1}^{4}(C_{x.k}X_{x.k}) = c_{1.1} \times x_{a.1} + c_{2.2} \times x_{n.2} + c_{3.1} \times x_{n.3} + c_{4.4} \times x_{i.4} + c_{5.4} \times x_{n.5} + c_{6.4} \times x_{i.6} = c_{1.1} \times 1 + c_{2.2} \times 1 + c_{3.1} \times 1 + c_{4.4} \times 1 + c_{5.4} \times 1 + c_{6.4} \times 1 = 33 + 42 + 23 + 35 + 44 + 17 = 194$. We obtained the optimized results (i.e. the allocations of the six users with the minimal cost). Here the best cumulative cost is equal to 194 and a total of 47 exchanged messages (see Tables 2 and 3).

Table 1: The costs table.

User \ Car park	1	2	3	4
1	33	71	77	∞
2	71	42	75	∞
3	23	82	45	∞
4	∞	52	55	35
5	∞	∞	∞	44
6	∞	∞	∞	17

Table 2: The candidate car park table.

user number	**user's variable**	$C_D(x)$	$C_D(x)_{\neq z}$
1	$x_{a.1}$	$\{1,2,3\}$	$\{\emptyset\}$
2	$x_{a.2}$	$\{1,2,3\}$	$\{\emptyset\}$
3	$x_{n.3}$	$\{1,2,3\}$	$\{\emptyset\}$
4	$x_{i.4}$	$\{4\}$	$\{2,3\}$
5	$x_{n.5}$	$\{4\}$	$\{\emptyset\}$
6	$x_{i.6}$	$\{4\}$	$\{\emptyset\}$

Table 3: The candidate car park table.

user number	user's variable	Allocated car park	Costs
1	$x_{a.1}$	1	$(c_{1.1} = 33)$
2	$x_{a.2}$	2	$(c_{2.4} = 42)$
3	$x_{n.3}$	1	$(c_{3.4} = 23)$
4	$x_{i.4}$	4	$(c_{4.2} = 35)$
5	$x_{n.5}$	4	$(c_{5.2} = 44)$
6	$x_{i.6}$	4	$(c_{6.4} = 17)$

3.5 Vehicles based Approach

The vehicles based approach consists in modeling all the vehicles as agents. Each agent holds four parameters: the agent's destination, the agent's user type, the user walking distance threshold, and the user's estimation occupation time (i.e. how much time does the user requests to occupy the parking space). The number of agents corresponds to the number of vehicles to monitor. We then map the following constraints:

The first constraint combines two equations (7), (8). Each single vehicle in the system must be assigned to exactly one car park, while respecting the capacity of each car park, those are structural constraints (hard constraints) and should be respected in any allocation in order to satisfy the consumers.

$$C_1(x) = \begin{cases} 0 & \text{if } O(X) \wedge C(X) \text{ } are\ true. \\ \infty & \text{otherwise.} \end{cases}$$

The $C_1(x) = \infty$ means that at least one of the two structural constraints are violated for the user x.

The second constraint is about the respect of the walking distance from a candidate car park for every single user. This constraint is needed to check that at least there is one eligible car park to make proposal for the user.

$$C_2(x) = \begin{cases} 0 & \text{if } C_D(x) \neq \emptyset \\ \infty & \text{otherwise.} \end{cases}$$

The $C_2(x) = \infty$ means that the system has no place to propose for the user x.

The third constraint favors the elderly and commuters and the sick over the ordinary user. This constraint serves to vehicles which move towards the same parking several times. Each time this constraint is violated, it will cost a unit. From the equation (10) we define the constraint $C_3(x_{i.k})$ as follows: $C_3(x_{i.k}) = \sum_{x' \in X_{\pi.k} \setminus x_{i.k}} \Delta(x_{i.k}, x')$.

The fourth constraint favors the vehicle which has an estimated travel time lower than that of its opponents this constraint finds its application between the permutations (x, x') of each subset of the set $X_{\pi.k}$ which are heading to the car park k. From the equation (12), we define the travel time constraint for the user x as follows: $C_4(x) = \sum_{x' \in X_{\pi.k} \setminus x} \Lambda(x, x')$.

The fifth constraint gives a priority to the user who is claiming to stay no longer than his competitor. This constraint finds its application between the permutations (x,x') of each subset of $X_{\pi.k}$ heading to the same parking slot k at the same time. From the equation (14), we define the fifth constraint $C_5(x)$ for the user x as follows: $C_5(x) = \sum_{x' \in X_{\pi.k} \setminus x} \rho(x,x')$.

The sixth constraint makes sure that when agents negotiate slots between different zones (i.e. when both $C_D(x)_{\neq z}$ and $C_D(x)$ are not empty) the cost is even lower for the user x.

$$C_6(x) = \begin{cases} 0 \;\; if \; C_D(x)_{\neq z} \; \wedge C_D(x) \; \neq \emptyset \wedge \sum_{j=1}^{5} C_j(x) \\ \leq \left[\sum_{j=1}^{5} C_j(x) \mid \forall p_k \in \{C_D(x) \setminus C_D(x)_{\neq z}\}\right] \\ 1 \; otherwise. \end{cases}$$

3.6 Car Park Approach

Instead of considering each vehicle as an agent, we can consider each car park as an agent. The number of agents corresponds to the number of car parks to monitor. We consider that every car park has the knowledge of all vehicles seeking to park in its zone. A car park agent holds an array variable T_k that contains the type, the trip time estimation and parking time of every vehicle heading to its zone. To model these vehicles, we use the six constraints of the previous approach.

3.7 Zone based Approach

In this approach, we consider that there is an agent per zone, where each agent zone has the knowledge of all vehicles on it . Thus the number of agents corresponds to the number of zones to monitor. We consider that each agent zone has the knowledge of all vehicles seeking to park on it. A zone agent contains an array variable T_z which contains the type, the trip time estimations and park duration estimations of each vehicle heading to it.

4 EXPERIMENTATION

4.1 The Choice of the Solver

The choice of the solver depends on the problem and the environment characteristics. DCOP algorithms can be classified as being either complete or incomplete, based on whether they can guarantee the optimal solution or they trade optimally for shorter execution times (Fioretto et al., 2018). In addition, DCOP algorithms are classified as synchronous or asynchronous (Farinelli et al., 2008). In our case and considering our constraints, we need a synchronous algorithm, since our constraints are strongly linked and need to be satisfied at the same time. We have conducted different experiments to determine the best solver for our model (among 12 solvers see Fig. 4). These allowed us to confirm that the Max-Sum algorithm is the fastest one in term of run times and number of messages generated followed by the DPOP algorithm. The Max-Sum is an incomplete, synchronous, inference-based algorithm based on belief propagation. It operates on factor graphs by performing a marginalization process of the cost functions, and optimizing the costs for each given variable. This process is performed by recursively propagating messages between variable nodes and factor nodes . The value assignments take into account their impact on the marginalized cost function. Max-Sum is guaranteed to converge to an optimal solution in acyclic graphs, but convergence is not guaranteed on cyclic graphs (Farinelli et al., 2008).

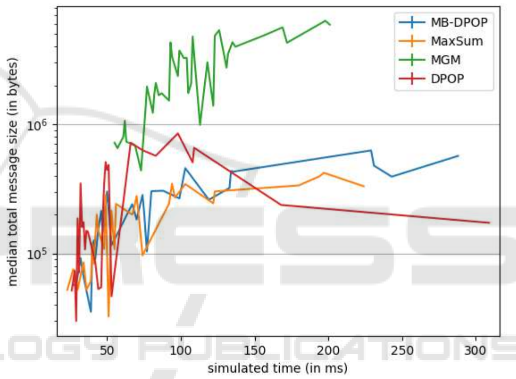

Figure 4: The illustration of the performances of the four most efficient solvers. Here we show the median total message size per simulated time (in ms) by each solver.

4.2 The Solution in the Dynamic Environment

Parking spaces during all day are either occupied or free, and their status changes in a volatile way over time and on a system needs to know the occupation status of each car park. Besides in the real environment vehicles continuously approach their destination, at each time step, we must define the new positions of the vehicles that take part in our DCOP problem, we mean the vehicles for which the DCOP will eventually maintain their previous allocations or revise certain according to the new updates in the environment. A way to cope with this issue is to simply solve the problem with the new current input, and consider starting a new instance only after the solver finishes executing the current instance. Our system needs a faster execution model by providing solutions of the static model more frequently. At every call of the solver the system might find a better allocation for already allocated vehicles, or provide a new allocation for vehicles that would have been left without allocation.

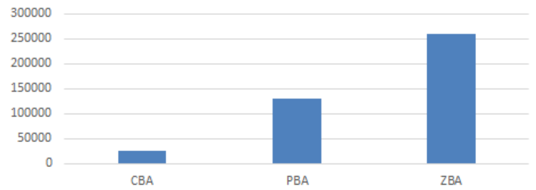

Figure 5: The sum of the simulation times per approach in (ms).

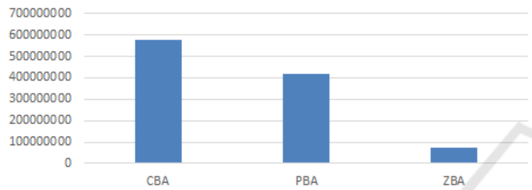

Figure 6: The total number of messages exchanged by approach (in bytes).

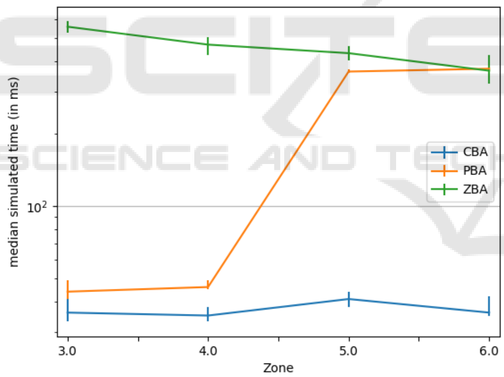

Figure 7: The median simulated time per zone (in ms) by approach.

4.3 Empirical Evaluation

In this section, we propose the Max-Sum algorithm as a resolver and frodo2 as a platform (Léauté et al., 2009), the experiments were performed using oracle VM virtual BOX with 24 GB RAM, under Linux 18.04(64 bit) with Intel core i5-7400 (six cores). We compare values from at least 1442 simulations, from 6 car parks to 16 car parks spread over [3,4,.., 6] zones per time. The range of requests varies on a domain [120,130, .., 180] per experimentation. We evaluate the performance of our mechanism on the three approaches described before. The first vehicle based

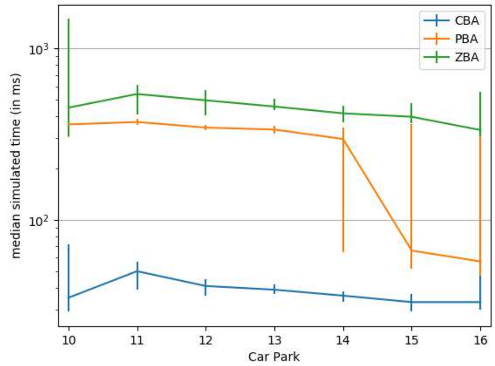

Figure 8: The median simulated time per number of car parks (in ms) by approach.

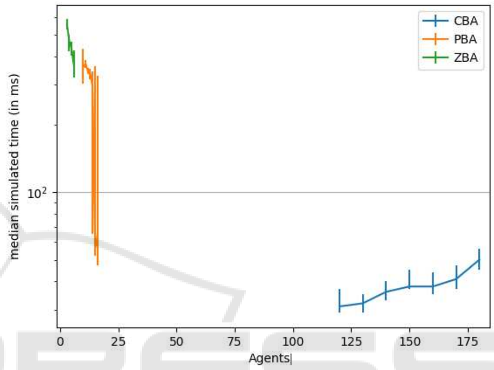

Figure 9: The median simulated time per number of agents (in ms) by approach.

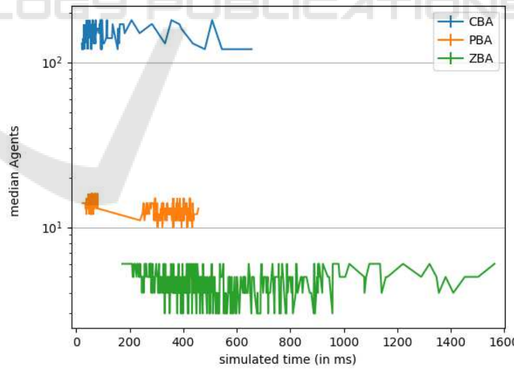

Figure 10: The median agent's number per simulated time (in ms) by approach.

approach is denoted by *CBA*. The second one, the car park based approach is denoted by *PBA*. The last one, the zone based approach is denoted by *ZBA*. From Fig. 5, we can infer that the fastest model is the model *CBA* approach followed by the *PBA* approach, the slowest one is the *ZBA* approach. From Fig. 6, we can see that the *CBA* exchanged the biggest volume data followed by the *PBA*. The smallest data is used by *ZBA*. From Fig. 7, it can be seen that the more the number of sectors (or zones) we use the more faster is the *ZBA* approach. On the other hand, *CBA* remains insensitive to the change of number of zones, however *PBA* becomes slower. From Fig. 8, we could say that as the number of car parks increases, the possibility of finding a parking space increases more. From Fig. 9 and Fig. 10, we can deduce that even though *CBA* approach is the fastest of the three approaches, the agents of the *CBA* start to run out of steam compared to the *PBA*, certainly because the number of agents is more important. From these approaches we can infer that the most suitable approach for the Cloud environment is the *PBA*. It is a model which is not the slowest amid the three approaches and which exchanges a volume of data which is not the highest.

5 CONCLUSION

As mentioned, our distributed model uses the VANETs that connect agents both for relaying the parking status and for inter-agent negotiations. The reservations are then relayed to a central server. By doing so, we decrease the volume of data exchanged in the Cloud. Contrarily, in the centralized case, there is a central server which gathers parking requests, real-time information (i.e., vehicle location, traffic condition, users preferences), etc. At every period of time T (related to the lifetime of the information on the availability of spots) the central server receives thousands of variables about the status of all car parks of the system, which could be excessively time-consuming and potentially incur high cost and slow-downs if there are huge amounts of data exchanged via the Cloud like in large car parks area. As a perspective, our next task will be to investigate the impact of our process on communication costs, and show that our model transfers less data and is therefore less expensive than the centralized one.

REFERENCES

Ayala, D., Wolfson, O., Xu, B., Dasgupta, B., and Lin, J. (2011). Parking slot assignment games. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 299–308.

Bessghaier, N., Zargayouna, M., and Balbo, F. (2012). Online localized resource allocation application to urban parking management. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 67–74. IEEE.

Boudali, I. and Ouada, M. B. (2017). Smart parking reservation system based on distributed multicriteria approach. *Applied Artificial Intelligence*, 31(5-6):518–537.

Chou, S.-Y., Arifin, Z., and Dewabharata, A. (2018). Optimal decision of reservation policy for private parking sharing system. In *2018 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, pages 66–71. IEEE.

Chou, S.-Y., Lin, S.-W., and Li, C.-C. (2008). Dynamic parking negotiation and guidance using an agent-based platform. *Expert Systems with applications*, 35(3):805–817.

Farinelli, A., Rogers, A., Petcu, A., and Jennings, N. R. (2008). Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, Volume 2*, pages 639–646.

Fioretto, F., Pontelli, E., and Yeoh, W. (2018). Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research*, 61:623–698.

Hedderich, M., Fastenrath, U., Isaac, G., and Bogenberger, K. (2017). Adapting the a* algorithm for park spot routing. *Transportation Research Procedia*, 27:1066–1073.

Kazi, S., Nuzhat, S., Nashrah, A., and Rameeza, Q. (2018). Smart parking system to reduce traffic congestion. In *2018 International Conference on Smart City and Emerging Technology (ICSCET)*, pages 1–4. IEEE.

Léauté, T., Ottens, B., and Faltings, B. (2010). Ensuring privacy through distributed computation in multiple-depot vehicle routing problems. In *Proceedings of the ECAI'10 Workshop on Artificial Intelligence and Logistics (AILog'10)*.

Léauté, T., Ottens, B., and Szymanek, R. (2009). Frodo 2.0: An open-source framework for distributed constraint optimization. In *Proceedings of the IJCAI'09 Distributed Constraint Reasoning Workshop (DCR'09)*, pages 160–164.

Picard, G., Balbo, F., and Boissier, O. (2018). Approches multiagents pour l'allocation de courses à une flotte de taxis autonomes. *Rev. d'Intelligence Artif.*, 32(2):223–247.

Weiss, M., Lubin, B., and Seuken, S. (2017). Sats: A universal spectrum auction test suite. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 51–59.

Zeenat, R., Ashifuddin, M., and Marijn, J. (2018). A multiple-criteria algorithm for smart parking: making fair and preferred parking reservations in smart cities. In *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, pages 1–9.