A Shapley Value-based Strategy for Resource Allocation in Vehicular Clouds

Aguimar Ribeiro Jr.
Universidade de São Paulo (ICMC/USP)
São Carlos, Brazil
aguimarjr@usp.br

Geraldo P. Rocha Filho
Universidade de Brasília (DCC/UnB)
Brasília, Brazil
geraldof@unb.br

Daniel L. Guidoni
Universidade Federal de Ouro Preto (DECOM/UFOP)
Ouro Preto, Brazil
guidoni@ufop.edu.br

Robson E. De Grande
Brock University (Dept. of Computer Science)
St. Catharines, Canada
rdegrande@brocku.ca

Sandra Sampaio
University of Manchester (Dept. of Computer Science)
Manchester, UK
s.sampaio@manchester.ac.uk

Rodolfo I. Meneguette
Universidade de São Paulo (ICMC/USP)
São Carlos, Brazil
meneguette@icmc.usp.br

***Abstract*—The continuous emergence of new applications for Internet-connected road vehicles is imposing unprecedented resource demand. Motivated by the incorporation of ever more resources into vehicles, this is a trend that, on the downside, is causing vehicular networks to become increasingly more challenging to manage. Departing from the proposition that computing capabilities can help overcome resource allocation problems in vehicular clouds (VCs), in this paper, we formulate ALTAIC, a coalition game to maximize resource utilization while dynamically load-balancing the usage among the VCs. First, we define a Shapley value-based strategy to determine the order in which the tasks are allocated. Then, with the marginal contribution of each task calculated, we employ a simple queue to allocate the tasks in VCs using these values. Finally, we conduct a comparative performance analysis of ALTAIC and relevant approaches. Simulation results show that the proposed solution allocates more tasks than the others and reduces 27.12% the load average of the VCs.**

***Index Terms*—VANET, vehicular clouds, resource allocation, load-balancing, game theory, Shapley value**

I. Introduction

The growth of the Internet of Things (IoT), its wide variety of connectable devices, and the incorporation of computational resources (e.g., storage, processing and bandwidth) into vehicles [1] have seen breakthrough innovations in vehicular network applications. This explosion of new applications has, nonetheless, brought new challenges, where efficient and effective allocation of computational resources for the fulfillment of application requirements is at the crux of them all. Thus, it has raised the following question: *how can computational resources be efficiently allocated to requesting applications whilst minimising resource idleness in vehicular networks* [2]*?* The search for an answer has brought about a paradigm shift from vehicular networks to vehicular clouds (VC*s*), where exploitation of computing capabilities through vehicular collaboration and resource sharing is the main objective [3] [4].

Since this shift, the bulk of research in vehicular networks has mainly focused on the following topics: *(i)* development of road traffic applications, *(ii)* management of VCs, and *(iii)* quality of services, including security and privacy. In the following, we describe earlier work of relevance to this paper. Considering topic *(i)*, Liu *et al.* [5] identify three main categories of applications: road safety (e.g., lowering the risk of accidents); traffic efficiency (e.g., reducing travel time and alleviating traffic congestion), and value-adding (e.g., providing infotainment, path planning, and internet access). In topic *(ii)*, a noteworthy work is the one by Paul *et al.* [6], which introduces the concept of vehicular cloud controller (VCT), an entity on the network edge responsible for creating and maintaining VCs as well as managing resources available from vehicles. Successful VCT operation depends on the communication between vehicles and roadside units (RSUs), where information, such as vehicle position, idle and required resources, is collected and provided to the VCT as vehicles move across different areas of the network. It is worth pointing out that vehicles are equipped with an onboard unit (OBU) and GPS for road map information, including the positions of RSUs.

In relation to topic *(iii)*, the early work by Olariu *et al.* [7] stands as one of the first to discuss security and privacy issues in connection with a variety of scenarios, where opportunities for idle resource exploitation are presented alongside possible applications. Whilst the described *static* scenarios involve sharing of resources from vehicles at long-term parking lots (e.g., large airports and shopping malls), and are mostly associated with event planning applications (e.g., sport events), more *dynamic* scenarios involve resources from vehicles trapped in traffic jams or stopped at traffic lights, and are typically associated with traffic light rescheduling, congestion dissipation, evacuation planning/management and on-road safety applications. In these scenarios, security and privacy issues revolve around authentication and identification of vehicles, trust between network nodes, node heterogeneity, and location and privacy of vehicles and their owners. Virtualization is suggested as the main mechanism for tackling these issues. Other concerns relate to architectural challenges that severely impact on the fulfillment of multiple and conflicting service requirements. For instance, reduced delays for real-time applications, high throughput and minimum bandwidth for radio channels in environments with many connected nodes, high mobility of vehicles, unstable network topologies, and limited resources.

Research in resource allocation in VCs is limited to load balancing techniques developed for static scenarios to mitigate delays in service delivery. Dynamic VC scenarios have been largely ignored due to the complexity associated with their modelling [8], particularly when trade-offs between multiple and conflicting service requirements need to be balanced, such as throughput and delay, resource utilization, energy consumption and security. VC resource allocation models that consider trade-offs are typically based on methods ranging in complexity from greedy algorithms [9] to reinforcement learning [10]. These models, however, fail to appropriately grasp the suitability of different strategic approaches to how data collected from multiple vehicles should be aggregated or combined to fulfill objectives, which is scenario-dependent. Such approaches can be, for example, collaborative, where vehicles cooperate to maximize a common objective, or competitive, where each individual vehicle tries to selfishly maximise its own objective. This limitation has a negative impact on trade-off decision-making, since it requires some level of interaction between participants and combination of their individual outcomes.

In this paper, we show that, where trade-offs need to be balanced and, thus, *interactive decision-making* is necessary, Game Theory is a better suited method than others suggested in the literature. Since it combines mathematical models to represent and solve complex optimization problems, it is better fitted to handle the complexities of highly dynamic and unstable scenarios [11], [12], which characterize vehicular networks.

We achieve this by proposing a novel metaheuristic algorithm - the shapLey-value based sTrAtegy in vehIcular Clouds (ALTAIC) - that models VCs as a coalition. More specifically, ALTAIC creates a collaborative environment that uses Shapley values to maximize resource utilization while dynamically load-balancing resource usage across VCs. The main contributions of this paper can be summarized as follows:

- Effective use of a Shapley value approximation method in vehicular network environments.
- A simpler resource allocation scheduler algorithm than others described in the literature.
- An efficient, cooperative, and dynamic load-balancing scheme across VCs considering resource utilization load average.
- A detailed demonstration that ALTAIC outperforms other, widely used, resource allocation algorithms by showing higher resource utilization under varying service request rates and dynamic load-balancing.

The rest of this paper is structured as follows. Section II reviews background material relevant to this paper. Section III presents recent related works. Section IV provides a detailed overview of target use case scenarios as well as ALTAIC's model and operating principle. Section V details the methodology and discusses the obtained results. Finally, Section VI summarizes achievements and points out future work directions.

II. Background

In this section, Coalitional Games (CG) and Shapley values are defined.

A. Coalitional Games

A CG is a typical cooperative game that aims to model situations where players may cooperate to achieve their goals [11]. It assumes players can form coalitions and engage in a binding agreement that yields a certain profit. The only requirement is that players be capable of arriving at decisions and committing to those decisions. The worth of a coalition is the maximal profit it can generate through the cooperation of its participants. Different disciplines extensively use CG theory and its theorems for modeling [13]. The definition of a CG is described as follows [11].

Definition 2.1 (Coalitional game): A CG is a pair (N, v) such that:

- $N = 1, 2, ..., n$ is a finite set of players. A subset of N is called a coalition. The collection of all the coalitions is denoted by 2^N.
- $v : 2^N \rightarrow \mathbb{R}$ is a function associating every coalition S with a real number $v(S)$, satisfying $v(\emptyset) = 0$. This function is called the coalitional function of the game.

A coalition formed by all players is named a grand coalition.

B. Shapley Values

Shapley value is a single-valued solution concept for coalitional games [14]. Considering the grand coalition N will be formed, the Shapley value answers how the worth $v(N)$ will be divided among the coalition members. The notion of a marginal contribution, the amount by which the player's participation increases the worth of a coalition, is central to the definition of Shapley value. An axiomatic approach defines the Shapley value: it is the unique solution concept that satisfies symmetry, null player, additivity, and efficiency properties [15]. **Symmetry**: interchangeable players i and j give the same marginal contribution to every coalition that does not contain them. Adding player i to a coalition is equivalent to adding player j to that coalition. **Null player**: if a player contributes nothing to any coalition, he is a null player and receives no part of the coalition's worth. **Additivity**: the worth of two independent games is the sum of the worth of each. **Efficiency**: when assumed that the coalition will form the grand coalition, its members entirely distribute the worth.

Let i be a player, and S be an arbitrary coalition that does not include player i, the Equation 1 gives the Shapley value of i, which is the sum of marginal contributions of i to every possible coalition formed by the permutation of the other players.

TABLE I
SUMMARY OF RELATED WORKS

Feature	[17]	[18]	[19]	ALTAIC
No pre-aggregated resources	✓		✓	✓
Variable request rates		✓		✓
Dynamic load-balancing				✓

$$\mathrm{Sh}_i(N;v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|! \times (n-|S|-1)!}{n!} (v(S \cup \{i\}) - v(S)) \quad (1)$$

Computing the exact Shapley values is generally $\mathcal{NP}$-hard, requiring approximation methods for non-trivial models [15]. It requires time $O(2^n)$ for n players, making it intractable for many players. So, in this work, we have employed a general approximation method that provides fast solutions with an acceptable margin of error [16].

III. Related Works

Various techniques for efficient resource allocation in vehicular networks have been proposed. For example, Costa *et al.* [17] presented a combinatorial optimization mechanism to allocate computational tasks in VCs. The objective was to select an optimal set of tasks to allocate in real-time to each VC, maximizing resource utilization. However, the study did not keep an eye on load-balancing and did not vary request rates. In [18], Yu *et al.* studied the sharing and management of bandwidth to mobile applications in vehicular cloud environments. They proposed a coalitional game model to support the cooperation between service providers to share their idle resources. It considered different service request rates, and the results showed a greater computational resource utilization above 75% than in scenarios without cooperation. Hattab *et al.* [19] formulated a model for task allocation as a combinatorial optimization problem to minimize the time for completing the task execution in the available VCs. The authors presumed the creation of a unique VC, and they did not vary request rates nor consider load-balancing during simulations.

As discussed in this section, only [18] utilizes different request rates during simulations. However, it assumes other resources than those shared by the vehicles, for example, from data centers. Furthermore, besides considering earlier clustered resources, it also does not assumes the existence of a load-balancing scheme. Hence, in our work, we propose ALTAIC. A heuristic solution modeling the resource allocation problem as a coalitional game, varying request rates during simulations and uses Shapley values to maximize resource utilization while dynamically load-balancing the usage among the VCs. Table I summarizes this section.

IV. System Model

A. System Overview

In an illustrative scenario, where there are RSUs deployed all over the map covering the whole road network area, the vehicles are grouped into VCs by running a clustering

Algorithm 1: ALTAIC Algorithm

```
Input: $T$ - set of all tasks, $VCs$ - set of VCs
$maxAvail \leftarrow$ Max(IdleResources($VCs$))
$tasks \leftarrow$ Filter($T, by =$ Weight $\leq maxAvail$)
$nExecs \leftarrow \gamma$
$eHistory \leftarrow \emptyset$
$aValues \leftarrow \emptyset$
$meanSV \leftarrow \emptyset$
$sortedTasks \leftarrow \emptyset$
for $i \leftarrow 0$ to $nExecs$ do
    $aValues \leftarrow$ CalcApproxSV($tasks$, vFunction)
    $eHistory \leftarrow eHistory \cup \{aValues\}$
$meanSV \leftarrow$ Mean($eHistory$)
$sortedTasks \leftarrow$ SortDesc($tasks, by = meanSV$)
TaskScheduler ($sortedSV, VCs$)
```

Algorithm 2: vFunction - Coalitional Function

```
Input: $T$ - sorted tasks to allocate, $VCs$ - set of VCs
Output: The worth generated by tasks $T$
$worth \leftarrow 0$
for task in T do
    $taskW \leftarrow$ Weight($task$)
    $availableVCs \leftarrow$ Filter($VCs, by =$ IdleResources $\geq$ $taskW$)
    if Len($availableVCs$) $= 0$ then
        continue
    $chosenVC \leftarrow$ Choice($availableVCs$)
    DecIdleResource ($chosenVC, taskweight$)
    $worth \leftarrow worth + taskweight$
return worth
```

algorithm every 60 seconds due to the high mobility of the vehicles. The available computational resources inside a VC can be shared and allocated by the VC Controller (VCT) to tasks waiting for resources to run.

There are x vehicles, where each vehicle denoted by $v_i : i \in [1, x]$ contains an OBU that allows communication among vehicles and between vehicles and RSUs. The RSUs are responsible for gathering the vehicles' information in real-time and forwarding the request for resources to the VCT. If a vehicle requires resources more significant than it can provide for running a task, it can request these extra resources from the VCT. The VCT is accountable for running the ALTAIC, which determines the best way to allocate tasks in each of the VCs to maximize resource utilization and load-balancing the requests according to the capacity of each VC.

B. ALTAIC

As shown in Algorithms 1, 2, and 3, we designed three algorithms to complete the entire resource allocation process in the VCs. Algorithm 1 is the main algorithm responsible for defining which cloud has its resources allocated to each task. Algorithm 2 is the coalitional function v that determines the worth of allocating a given task set and a group of VCs. The Shapley value approximation method mentioned in the Introduction uses that function v to get Shapley values according to Equation 1. Finally, Algorithm 3 is the scheduling algorithm.

Firstly, Algorithm 1 needs to determine which VC has the most available resources (line 1) because it is not feasible to allocate tasks weightier than that; the tasks are considered indivisible for this work, so they are filtered out (line 2). Next, the algorithm initializes the auxiliary variables (lines 3-7) and calculates the Shapley value approximations for the remaining tasks γ times (lines 8-10). Then it calculates the means of the

Algorithm 3: TASK SCHEDULER Algorithm

```
Input: $T$ - sorted tasks to allocate, $VCs$ - set of VCs
Output: The worth generated by the allocated tasks $T$
1  $worth \leftarrow 0$
2  for task in T do
3      $taskW \leftarrow$ Weight($task$)
4      $availableVCs \leftarrow$ Filter($VCs, by =$ IdleResources $\geq taskW$)
5      if Len($availableVCs$) $= 0$ then
6          continue
7      $chosenVC \leftarrow$ Choice($availableVCs$)
8      DecIdleResource ($chosenVC, taskweight$)
9      $worth \leftarrow worth + taskweight$
10     Allocate ($task, to = chosenVC$)
11 return worth
```

approximations (line 11) obtained in each iteration and uses these values to sort the tasks (line 12) in descending order (higher marginal contributions to the final worth). Finally, the algorithm calls the scheduler passing the sorted task set and the VCs (line 13).

In Algorithm 2, the worth of a task set is computed. It initializes the auxiliary variable (line 1), then iterates over each task (lines 2-9) and, at each iteration, determines which VCs have enough idle resources (lines 3-4). Every VC with enough resources is eligible to allocate the task. After that, the function picks one of them randomly (line 7), decreases the VC's idle resource counter (line 8), increases the worth (line 9), and repeats the process until there are no tasks (line 2) nor available resources (lines 5-6) and returns the worth of the task set (line 10). No allocation is done at this moment. Following Equation 1, the task set passed to the function v can be formed by any permutation of tasks. Finally, Algorithm 3 receives the list of tasks sorted by the Shapley values and allocates them (line 10), observing the order. The worth of the task set is returned (line 11). The algorithm is similar to Algorithm 2, except the task set contains all the tasks, and the allocation happens.

V. PERFORMANCE EVALUATION

This section presents the methodology to evaluate ALTAIC's proposed model. The experiments have used the Simulation of Urban MObility (SUMO) [20], version 1.12.0, joined with Luxemburg's mobility trace [21]. That trace contains 24h of vehicular mobility with heights of up to five thousand vehicles. The algorithms were implemented in Python using the Traffic Control Interface (TraCI) library to connect to SUMO. We have decided to evaluate ALTAIC during the interval between 11a.m and noon due to the lower vehicular density on the map, where fewer resources provide challenging environments.

This work also assumed that since VCT allocates the resources to the task, it has all it needs to execute and complete itself, and its deadline (system runtime) is equal to 1. In addition, since the high mobility in the scenario, VCT runs DBSCAN [22] every 60 seconds to create the VCs where each vehicle can share up to 3 resource units. The tasks' arrival rate follows Poisson distributions of different average request rates in $\{20, 50, 100\}$, and the tasks' weight follows uniform distributions ranging from $[1, \mu]$, where $\mu = \{20, 25, 30, 40, 50, 75, 100\}$. Table III shows the simulation parameters.

TABLE II
SIMULATION PARAMETERS

Parameter	Value
Communication range	100 meters
Task arrival mode	Poisson distribution
Request rates (λ)	20, 50, 100
Task weights	20, 25, 30, 40, 50, 75, 100
Simulation time	1h Lust Scenario (Interval: 11-12h)
Clustering algorithm	DBSCAN $\rightarrow$ eps = 100, min_samples = 2
Clustering interval	60 seconds
Number of vehicles	1000 $\sim$1500
Resources by vehicle	1, 2, 3

TABLE III
NOTATION DEFINITIONS

Variable	Definition
VC	Set of all available VCs
UVC	Set of VCs which allocated at least one task
T	Set of all tasks
T^A	Set of allocated tasks
X_i	Element i in the set X
$weight(T)$	Number of resources needed by task T
$resource(VC)$	Available resources in VC

To evaluate the solution, we considered the notations listed in Table III, denoting *UVC* as the set of VCs effectively used for allocating at least one task and *AT* as the percentage of allocated tasks in a scenario. They are defined as follows:

$$UVC = \bigcup_{i=0}^{|VC|} VC_i \times k \begin{cases} k = 1 & VC_i \text{ is used} \\ k = \emptyset & VC_i \text{ is not used} \end{cases} \tag{2}$$

$$AT(\%) = \frac{\sum_{i=0}^{|T^A|} weight(T_i^A)}{\sum_{j=0}^{|T|} weight(T_j)} \cdot 100 \tag{3}$$

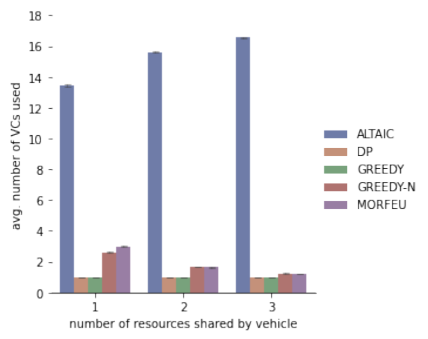
In addition, we denoted *LA* as the load average resource usage of the VCs in *UVC* and *NTL* as the ratio between the total resources needed to meet all service requirements and the available resources in VCs. Finally, *LAN* as the weighted load average resource usage of the VCs in *UVC* by *NTL*. They are defined as follows:

$$LA(\%) = \frac{\sum_{i=0}^{|T^A|} weight(T_i^A)}{\sum_{j=0}^{|UVC|} resource(UVC_j)} \cdot 100 \tag{4}$$

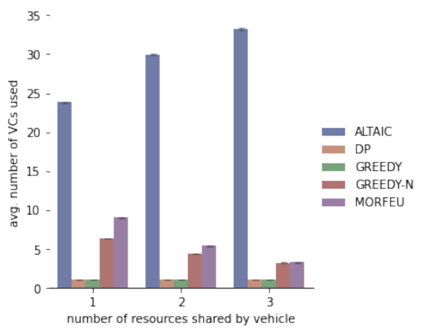
$$NTL = \frac{\sum_{i=0}^{|T|} weight(T_i)}{\sum_{j=0}^{|VC|} resource(VC_j)} \cdot 100 \tag{5}$$

$$LAN(\%) = \frac{LA}{NTL} \cdot 100 \tag{6}$$

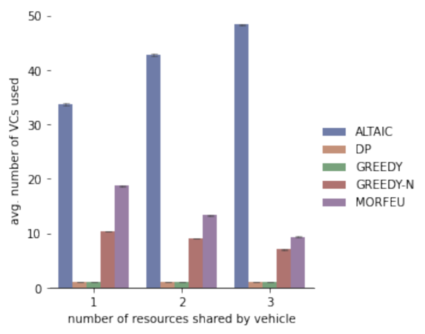
We considered four other mechanisms to analyze their performance against ALTAIC in allocating tasks and load-balancing in a VC environment: MORFEU [17], GREEDY-N based on Nabi *et al.* [23], DP, and GREEDY. The first two methods consider all the existing VCs, while the others only assume one VC. MORFEU considers a combinatorial optimization-based task allocation as introduced in Section III.



(a) Request rate = 20



(b) Request rate = 50



(c) Request rate = 100

Fig. 1. Average number of VCs used (UVC) under scenarios with different request rates.

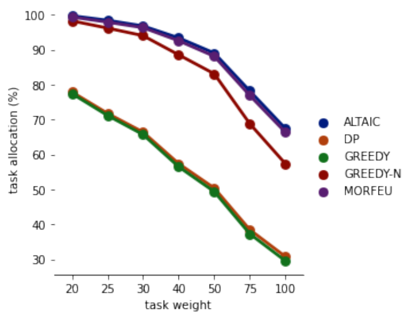


Fig. 2. Overall task allocation by task weights.

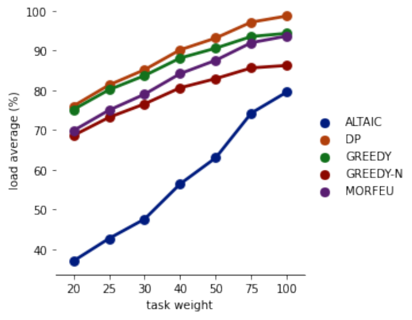


Fig. 3. VCs' load average by task weights.

On the other hand, the DP solution considers a dynamic programming approach, and GREEDY e GREEDY-N are greedy algorithms as they are named after. All three algorithms sort tasks in descending manner by their weights and first utilize the VC with more resources to begin the allocation.

A. Simulation Results

In this section, we evaluate the performance of the proposed resource allocation strategy. The simulation results show the performance under the comparison of the five methods with a confidence interval of 95%.

Figure 1 shows the average number of VCs engaged in resource allocation ($|UVC|$) under scenarios with different request rates and vehicles sharing one, two, or three resource units each. DP and GREEDY use only one VC by definition so that it does not vary over the scenarios. With the increase in request rates for all the other algorithms, although there is an increase in the number of VCs used, our proposed solution can significantly spread tasks out over a higher number of VCs. ALTAIC uses at least 346.67%, 164.44%, and 80.21% more VCs, respectively, in Figures 1(a), 1(b), and 1(c). It should be noted that even when the number of resources shared by vehicles increases, creating environments with abundant resources, ALTAIC uses more VCs to allocate tasks, differently from the other methods, which decreases the number of VCs used. The coalition formed by VCs with abundant resources relaxes the constraint of the defined coalitional function (Algorithm 2 at line 4), which gives room for greater Shapley values according to Equation 1.

Figure 2 shows the overall task allocation (AT) under different scenarios. Our algorithm performs better than DP, GREEDY, GREEDY-N, and MORFEU, respectively 32.84%, 33.7%, 5.24%, and 0.76% on average. However, with the increase in tasks' weight, the performance of all methods decreases as expected since the tasks are considered not divisible, and a single VC cannot meet more demanding task requirements.

Figure 3 shows the VCs' load average resource usage (LA). Again, ALTAIC outperforms the other algorithms significantly with lower usage of resources. Our method can allocate more tasks (Fig. 2), spreading them out to a more significant number of VCs (Fig. 1) which distribute the load among the VCs, avoiding their capability saturation. On average, a 27.12% reduction in LA is observed. This energy and performance-aware load-balancing strategy can achieve balanced energy consumption, reduce delay and improve network utilization in heterogeneous vehicular networks [24].

Figure 4 shows the VCs' weighted load average resource usage (LAN) evolution over the increasing volume of resources requested by the tasks and the total available resources in the network (NTL). Compared with the other methods, which abruptly raise the usage of resources even with lower NTL, ALTAIC slowly increases LAN, keeping lower levels until NTL reaches 80%. At this point, where the service requests are demanding up to 80% of all available resources, ALTAIC stabilizes load average slightly above the GREEDY-N's. However, as shown in Figure 5, when the demand for resources exceeds 80%, ALTAIC outperforms GREEDY-N

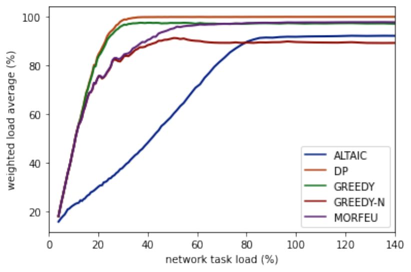


Fig. 4. VCs' weighted load average by task load in the network.

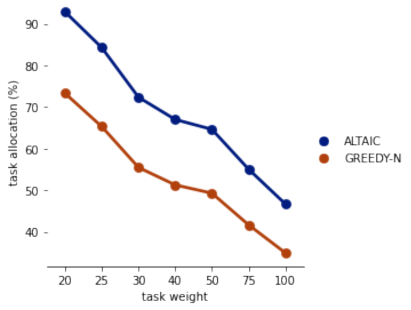


Fig. 5. Task allocation by task weights when demand exceeds 80% of the total available resources in the network.

with task allocation rates of 15.94% higher on average. Our proposed algorithm can accurately solve the problem with higher efficiency and a better load-balancing strategy.

VI. Conclusion

In this paper, we proposed ALTAIC, a heuristic algorithm that organizes vehicular clouds as a grand coalition and uses Shapley values to maximize resource utilization while dynamically load-balancing resource usage across VCs. Simulation results show that ALTAIC fulfills its objectives by providing resource utilization with a better load average across VCs than that of widely used algorithms in the literature. In addition, ALTAIC'S task scheduling is simpler, more intuitive and straightforward as it follows the descending order of the Shapley value attributed to each task. In other words, it first allocates resources to tasks with higher Shapley value. We believe the combination of simplicity and efficiency of ALTAIC will pave the way to a new generation of Cloud resource management technologies.

Future work includes forming smaller coalition structures that are more beneficial than a grand coalition when and if they exist. Likewise, examining incentive mechanisms for a VC does not obtain more gains by violating the suggested coalitions.

Acknowledgment

This work was partially supported by FAPESP (São Paulo Research Foundation) under grant #2020/07162-0, #2022/00660-0, #2020/05126-6, and #2021/06210-3.

References

[1] U. Cisco, "Cisco annual internet report (2018–2023) white paper," https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html, 2020, accessed on: Nov. 4th, 2021.

[2] W. Wei, R. Yang, H. Gu, W. Zhao, C. Chen, and S. Wan, "Multi-objective optimization for resource allocation in vehicular cloud computing networks," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[3] M. Abuelela and S. Olariu, "Taking vanet to the clouds," in *Proceedings of the 8th international conference on advances in mobile computing and multimedia*, 2010, pp. 6–13.

[4] M. Eltoweissy, S. Olariu, and M. Younis, "Towards autonomous vehicular clouds," in *International Conference on Ad hoc networks*. Springer, 2010, pp. 1–16.

[5] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular edge computing and networking: A survey," *Mobile networks and applications*, vol. 26, no. 3, pp. 1145–1168, 2021.

[6] A. Paul, N. Chilamkurti, A. Daniel, and S. Rho, *Intelligent vehicular networks and communications: fundamentals, architectures and solutions*. Elsevier, 2016.

[7] S. Olariu, T. Hristov, and G. Yan, "The next paradigm shift: from vehicular networks to vehicular clouds," *Mobile ad hoc networking: cutting edge directions*, pp. 645–700, 2013.

[8] A. R. Hameed, S. ul Islam, I. Ahmad, and K. Munir, "Energy-and performance-aware load-balancing in vehicular fog computing," *Sustainable Computing: Informatics and Systems*, vol. 30, p. 100454, 2021.

[9] W. Yang, R. Zhang, C. Chen, and X. Cheng, "Secrecy-based resource allocation for vehicular communication networks with outdated csi," in *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*. IEEE, 2017, pp. 1–5.

[10] H. Peng and X. Shen, "Deep reinforcement learning based resource management for multi-access edge computing in vehicular networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2416–2428, 2020.

[11] S. Zamir, M. Maschler, and E. Solan, *Game theory*. Cambridge University Press, 2020.

[12] Z. Sun, Y. Liu, J. Wang, G. Li, C. Anil, K. Li, X. Guo, G. Sun, D. Tian, and D. Cao, "Applications of game theory in vehicular networks: A survey," *IEEE Communications Surveys & Tutorials*, 2021.

[13] A. Safdarian, P. H. Divshali, M. Baranauskas, A. Keski-Koukkari, and A. Kulmala, "Coalitional game theory based value sharing in energy communities," *IEEE Access*, vol. 9, pp. 78 266–78 275, 2021.

[14] L. S. Shapley, "A value for n-person games, contributions to the theory of games, 2, 307–317," 1953.

[15] R. Mitchell, J. Cooper, E. Frank, and G. Holmes, "Sampling permutations for shapley value estimation," 2022.

[16] K. Corder and K. Decker, "Shapley value approximation with divisive clustering," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, 2019, pp. 234–239.

[17] J. B. D. da Costa, M. L. M. Peixoto, R. I. Meneguette, D. L. Rosário, and L. A. Villas, "Morfeu: Mecanismo baseado em otimizaçao combinatória para alocaçao de tarefas em nuvens veiculares," in *Anais do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. SBC, 2020, pp. 505–518.

[18] R. Yu, X. Huang, J. Kang, J. Ding, S. Maharjan, S. Gjessing, and Y. Zhang, "Cooperative resource management in cloud-enabled vehicular networks," *IEEE Transactions on industrial electronics*, vol. 62, no. 12, pp. 7938–7951, 2015.

[19] G. Hattab, S. Ucar, T. Higuchi, O. Altintas, F. Dressler, and D. Cabric, "Optimized assignment of computational tasks in vehicular micro clouds," in *Proceedings of the 2nd International Workshop on Edge Systems, Analytics and Networking*, 2019, pp. 1–6.

[20] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[21] L. Codecá, R. Frank, S. Faye, and T. Engel, "Luxembourg sumo traffic (lust) scenario: Traffic demand evaluation," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 2, pp. 52–63, 2017.

[22] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[23] M. Nabi, R. Benkoczi, S. Abdelhamid, and H. S. Hassanein, "Resource assignment in vehicular clouds," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[24] A. R. Hameed, S. ul Islam, I. Ahmad, and K. Munir, "Energy-and performance-aware load-balancing in vehicular fog computing," *Sustainable Computing: Informatics and Systems*, vol. 30, p. 100454, 2021.