Computation Offloading and Retrieval for Vehicular Edge Computing: Algorithms, Models, and Classification

AZZEDINE BOUKERCHE and VICTOR SOTO, University of Ottawa

The rapid evolution of mobile devices, their applications, and the amount of data generated by them causes a significant increase in bandwidth consumption and congestions in the network core. Edge Computing offers a solution to these performance drawbacks by extending the cloud paradigm to the edge of the network using capable nodes of processing compute-intensive tasks. In the recent years, vehicular edge computing has emerged for supporting mobile applications. Such paradigm relies on vehicles as edge node devices for providing storage, computation, and bandwidth resources for resource-constrained mobile applications. In this article, we study the challenges of computation offloading for vehicular edge computing. We propose a new classification for the better understanding of the literature designing vehicular edge computing. We propose a taxonomy to classify partitioning solutions in filter-based and automatic techniques; scheduling is separated in adaptive, social-based, and deadline-sensitive methods, and finally data retrieval is organized in secure, distance, mobility prediction, and social-based procedures. By reviewing and analyzing literature, we found that vehicular edge computing is feasible and a viable option to address the increasing volume of data traffic. Moreover, we discuss the open challenges and future directions that must be addressed towards efficient and effective computation offloading and retrieval from mobile users to vehicular edge computing.

CCS Concepts: • **Information systems** → **Mobile information processing systems**; *Retrieval efficiency*; • **Networks** → *Cloud computing*;

Additional Key Words and Phrases: Computation offloading, vehicular cloud, data retrieval

ACM Reference format:
Azzedine Boukerche and Victor Soto. 2020. Computation Offloading and Retrieval for Vehicular Edge Computing: Algorithms, Models, and Classification. *ACM Comput. Surv.* 53, 4, Article 80 (August 2020), 35 pages.
https://doi.org/10.1145/3392064

1 INTRODUCTION

The proliferation of smart mobile devices and the increased demand for computing intensive application in these devices will lead to huge amount of data offloading to the cloud when using traditional mobile cloud computing (MCC) paradigm. This approach results in drawbacks for the system such as poor performance for the mobile application and network congestion, which leads to communication delay, packet loss, and blocking of new connections [Yao et al. 2015]. In this context, Mobile Edge Computing (MEC) [Mach and Becvar 2017] emerges as a viable solution to reduce the amount of data to be transferred for cloud servers. Computation in the edge takes advantage of nodes, like traffic light controllers or Road-side Units (RSUs), to alleviate the load from

Authors' address: A. Boukerche and V. Soto, EECS, University of Ottawa, K1N 6N5, Canada; emails: boukerch@eecs.ottawa.ca, vsoto101@uottawa.ca.

0360-0300/2020/08-ART80 $15.00
https://doi.org/10.1145/3392064

the cloud servers in the core layer. By bringing the processing nodes to the edge (i.e., closer to the user) communication times are decreased. Therefore, computation cost decreases making the CO technique a more efficient one. Moreover, as data can be analyzed at the edge, network traffic to a data center or cloud is reduced as well.

Recently, vehicular edge computing (VEC) has been proposed thanks to the considerable improvements on vehicles' storage, communication and processing power. The VEC paradigm proposes the use of vehicles as infrastructure for supporting processing of resource-constrained mobile devices' applications [Feng et al. 2017]. Therefore, the VEC paradigm can significantly alleviate network congestion in the core layer and reduce the overall cost of computation offloading by bringing the processing nodes closer to the user. VEC is similar to MEC in the sense of leveraging resources in the edge of the network, rather in the network core and cloud servers, for supporting resource-constrained devices' mobile applications. However, VEC proposed to explore vehicles as edge devices, which has more powerful resources as compared to traditional edge devices (e.g., cloudlets and RSUs) but incurs in unique and daunting research challenges that needs to be tackled [Sun et al. 2019].

The primary challenge in VEC is related to the mobility and connectivity of the edge nodes, i.e., vehicles. Data and computation offloading were extensively studied in the literature for cloud and MEC scenarios [Mach and Becvar 2017]. However, the use of vehicles as edge nodes brings novel critical challenges for data and computation offloading tasks, such as edge resource discovery, connectivity management for computation offloading for highly mobile vehicle edge nodes, and result retrieval. Those challenges mainly emerge from the high mobility, dynamic connectivity, and topology of vehicles. Hence, offloading data to an edge node with such high-mobility as vehicles becomes an issue at the time of sending the packets. Moreover, by the time the vehicle finishes processing the assigned task, it will want to return the output to the sender. Depending on some parameters, such as the speed and the processing time of the task, the vehicle's position might be different from the position it occupied when the task was received.

Many works in the literature have extensively covered mobile cloud and edge computing from a different perspective (e.g., computation offloading, resource management, energy efficiency, virtual machine migration, and offloading decisions). For instance, Mastelic et al. [2014] shed light on the high energy cost aspect for supporting cloud computing systems in data centers. Toosi et al. [2014] addressed the cloud interoperability critical challenging. They proposed a taxonomy to better highlight the challenges for inter-cloud operability, as well as to discuss the research works for inter-operable clouds proposed in the literature. Medina and García [2014] covered the research regarding virtual machine replication and migration in cloud servers, aimed at guaranteeing high availability of provided services. Samant and Bellur [2016] discussed works in the literature that proposed solutions for boosting the large-scale deployment of virtual machines in data centers. In addition, there are different approaches that have been taken for classifying the process of computation offloading [Akherfi et al. 2018; Orsini et al. 2015]. Moreover, works which survey MEC [Hirsch et al. 2018; Roman et al. 2018] focus on techniques that aim to process data in the cloud servers, and while it is a proven method, the size of data offloaded to the cloud continues to increase. Therefore, a solution as VEC becomes necessary to address the challenges of network congestion, which leads to delay, packet loss and blocking of new connections. However, it is essential to reveal the relationship between connectivity, communication capability, and mobility, as well as the benefits of exploiting vehicular resources for computation offloading in Vehicular Edge Computing.

This survey is aimed at filling the gap not addressed in the above-mentioned works. It tackles the scenario where vehicles are used as edge nodes for supporting computation-intensive applications from resource-constrained mobile devices. This scenario incurs in several new challenges for computation offloading and result retrieval, due to the mobile nature of the edge servers (i.e., vehicles). In this article, we survey proposed solutions for computation offloading in vehicular edge. We propose to classify task partition techniques in filter-based and automatic, based on the parameters each technique aims to improve. Moreover, we classify scheduling algorithms as adaptive, social-based, and deadline-sensitive, based on the environment for which the methods are considered . Finally, the data retrieval techniques are classified in distance-based, social-based, and mobility-prediction-based algorithms, based the distance references that the methods used to achieve an efficient retrieval. The remainder of this article is organized as follows: Section 2 discusses the twofold concepts and characteristics of vehicular edge computing and vehicular edge computing networks. A background to computation offloading, the process it follows and existing literature on platforms that take advantage of the CO technique is presented in Section 3. The classification proposed of existing techniques for partitioning, scheduling, and data retrieval for CO in a VEC environment are discussed in Sections 5, 6, and 7, respectively. Furthermore, existing techniques that can be adapted for computation offloading in a vehicular environment are discussed in Section 8. Finally, Section 9 gives the concluding remarks.

2 VEHICULAR EDGE COMPUTING AND VEHICULAR EDGE COMPUTING NETWORKS

Overall, vehicular edge computing proposes to offload tasks from resource-limited devices to be processed on nearby vehicles. This contributes to reduce the energy consumption at the resource-constrained devices and alleviate the traffic on the network backbone. In contrast, it is worth mentioning that vehicles will be equipped with different advanced onboard sensors, controllers and actuators, as well as will be required to perform computation-intensive tasks related to vehicular sensor networking applications for smart cities, multimedia-enriched applications for in-car experiences, sensing, processing, and actuation for smart transportation applications, and computing-intensive tasks for autonomous driving [Boukerche and Coutinho 2019a].

In this article, it important to highlight that this survey is aimed to cover the works in the area of vehicular edge computing. More specifically, it will discuss works in the literature that have proposed solutions for the problem of task offloading from mobile resource-constrained devices to edge computing infrastructures composed of vehicles. However, for the sake of completeness, in the following, we discuss the motivation and the main challenges for vehicular edge computing networks, as well as highlight recent works proposed in the literature to cope with the discussed challenges.

The expected tasks mentioned above will demand resources that will go far beyond the capabilities of a single intelligent vehicle. Therefore, computationally intensive tasks, or part of them, will be offloaded to cloud servers. The main disadvantage of this approach is the increased delay for task offloading, especially given in the scenario of vehicular networks, where vehicles will be moving and frequently suffering from disconnections to the cloud. Wide area networks with long and ultralong communication links can be used to guarantee the vehicle will always be able to communicate with the cloud for task offloading and results retrieval. However, such links will present low reliability, which will increase retransmissions, delay, and network congestion and overhead.

In recent years, mobile edge computing (MEC) [Abbas et al. 2018] has proposed to provide communication, processing, and storage capabilities in close proximity to mobile users. MEC can be implemented on 5G small-cells [Coutinho and Boukerche 2020; Ning et al. 2019b] and Road-side Units (RSUs) [Feng et al. 2019] and used to process offloaded tasked by intelligent vehicles, in order to reduce delay, network overhead, and congestion, as well as energy cost [Ning et al. 2019a]. Hence, vehicles can offload their compute-intensive tasks to MEC infrastructures. The MEC infrastructure will then offload tasks to a cloud server when it does not have enough resources to process it.

One of the main challenges in vehicular edge computing networks is to guarantee the vehicular connectivity with the MEC infrastructure. In order to maximize the communication opportunity, some works in the literature have proposed to use the predicted information of link duration to select the appropriate MEC for task offloading [Zhang et al. 2017]. Moreover, the duration of a link between a vehicle and a MEC infrastructure might not be enough to guarantee that the vehicle will still be connected to the MEC and capable of receiving the result after the processing completion of the offloaded task. In this regard, delay-sensitive tasks must be offloaded to the cloud for fast processing. However, the communication link between the MEC and the cloud will be the bottleneck, in terms of latency, for intelligent vehicular applications.

In this regard, works in the literature have proposed novel architectures aimed at efficiently interconnecting MEC and cloud infrastructures to reduce the communication delay. For instance, Wu and Zheng [2020] studied the uplink local delay considering a MEC-based VANET highway application, where the carrier sense multiple access (CSMA) mechanism is considered for channel access. Guo et al. [2019] introduced the fiber-wireless (FiWi) integration in order to improve vehicular edge computing network applications by reducing the latency of MEC and cloud communication. A FiWi architecture is also considered in Zhang et al. [2019]. However, the authors proposed a software-defined networking architecture for a load-balanced task offloading among MEC servers aimed at reducing latency.

3 COMPUTATION OFFLOADING IN A NUTSHELL

This section highlights the main building blocks involved in the computation offloading process from resource-constrained mobile devices to vehicular edge computing. Before we proceed further, it is worth to mention that the discussion provided in this section is upon the decision that a given task will be offloaded by the mobile device. Many works in the literature discuss the characteristics a task might have to be able to be offloaded. Moreover, tasks can be partially or entirely offloaded [Bozorgchenani et al. 2017; Kuang et al. 2019], where such decisions can be made by considering several factors, such as energy, delay, bandwidth, processing time, available resources, and cost [Boukerche et al. 2018; Boukerche et al. 2019]. In this article, we discuss the computation offloading tasks under the following assumptions. First, a decision has been made at the mobile device favorable to the computation offloading to a vehicular edge computing system. Second, the task is entirely offloaded. In this regard, this section highlights the three main steps for computation offloading of a computation-intensive task. After that, we discuss the potential of vehicular edge computing for reducing computation offloading for cloud servers and, consequently, network overhead.

With the development of the technology of mobile devices, applications are evolving into the creation of more and more compute-intensive applications to give the user a better experience. For example, in the case of gaming and social media, devices are capable of rendering video streaming with such good quality graphics. While all of these applications may improve the user experience, there is a disadvantage to their execution; the high consumption of resources on the mobile devices side. Moreover, the increasing usage of mobile devices presents the possibility of an overload in the bandwidth of the network. Devices using the computation offloading model and sending significant amounts of data to the core layer of the network can create bandwidth congestion. Mobile Edge Computing (MEC) extends the cloud paradigm by processing data in edge nodes, which are located at the edge of the network as shown in Figure 1. This approach alleviates the congestion in the bandwidth and at the same time reduces the communication cost by reducing the delay in the data retrieval process.

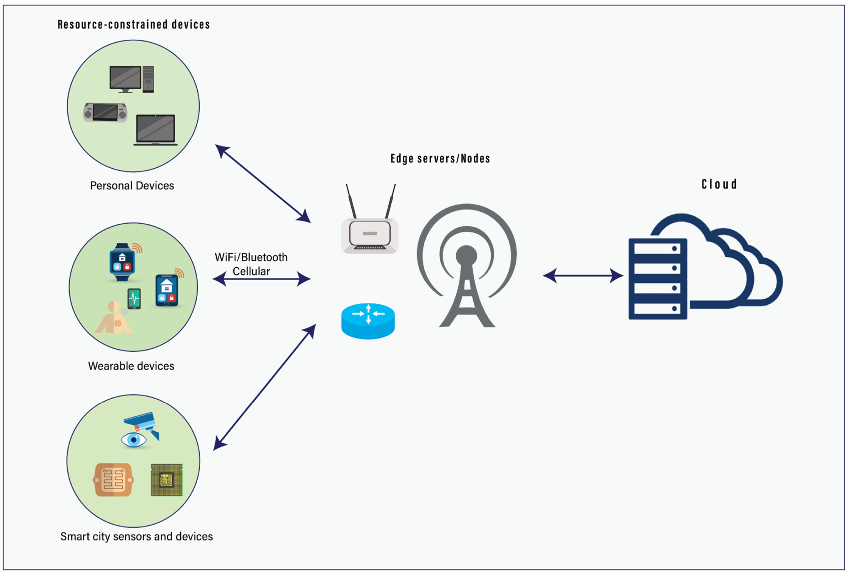

Fig. 1. Mobile Edge Computing architecture.

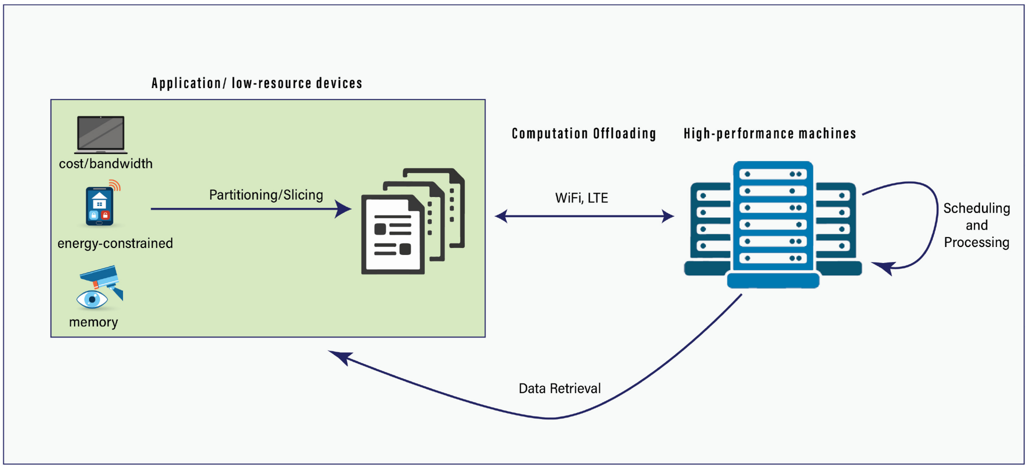

Fig. 2. Typical computation offloading architecture.

The computation offloading problem for MEC and cloud architectures has been extensively studied in the literature. Herein, however, this problem is addressed from a novel point of view, which considers that offloaded tasks will be executed on mobile vehicles as edge nodes, rather than traditional edge nodes and cloud servers. The use of vehicles as edge nodes brings additional benefits, such as the availability of powerful storage and computing resources, as well as their proximity to the resource-constrained mobile users, which does not incur in network core congestions. However, this approach presents new daunting challenges that need to be tackled, such as the vehicular mobility and connectivity management for computation offloading and results' retrieval. Those aspects are discussed in the following sections.

Figure 2 shows the main components and process of computation offloading, where instead of completely executing the application locally, it allows the device to transfer part of the tasks to a remote server. This approach aims to achieve a reduced execution time and energy consumption from the overall process.

The major steps for computation offloading in a vehicular edge computing environment, represented in Figure 2, are regarded as follows:

(1) *Partitioning*. The application is sliced into smaller offloadable tasks, usually on the mobile device. This step of the process may seem unaffected by the fact that a high-mobility environment is considered. However, if the partitioning is done efficiently according to the constant changing of nodes and resources, the following steps can have a better performance.
(2) *Scheduling*. The server considers the available resources and policies to process the offloaded tasks in the most efficient way. Considering the mobility of nodes in the case of a vehicular edge computing system, resource allocation becomes a more complex process. Therefore, a scheduling algorithm that can adapt to such environment can result in an enhanced QoS.
(3) *Data Retrieval*, where the output processed data is collected by the mobile device. The fact that the offloading is in an environment with high-mobility makes this step an essential one. While the two previous steps can help to achieve a quicker task execution, there are still challenges to consider for the retrieval, such as intermittent connectivity and message redundancy leading to a broadcast storm.

Based on the MEC architecture (Figure 1) and following the process of computation offloading (Figure 2) it is a viable option to consider vehicles as high-performance machines, which are most of the time idle and have unused resources. Compared to MEC, using vehicles as support infrastructure in an edge computing (VEC) is less costly, considering there is no need for deployment. Moreover, considering the capability of vehicles to communicate with RSUs, which are at the edge of the network, using vehicles as processing nodes greatly improves resource utilization and performance of MEC by reducing delay and capacity limitations when data traffic rises to create network congestions.

3.1 Computation Offloading in VEC

Vehicular edge computing aims to alleviate the overload from the core cloud servers and lower the offloading cost of mobile applications [Vigneri et al. 2017]. Figure 3 represents a typical architecture for VEC, where the resources of vehicles are exploited as edge nodes. The three entities of a typical VEC architecture can be explained as follows:

- *Mobile Devices:* comprise the resource-constrained device where applications being executed by users.
- *Roadside Units:* represent the fixed infrastructure of the network that receive and process the tasks offloaded by the mobile devices.
- *Vehicular Edge Nodes:* is formed by vehicles with powerful resources capabilities, which are able of communicating with each other and an RSU.

The objective of VEC is for resource-constrained mobile devices to offload applications to vehicular nodes in the edge of the network, in order to alleviate the congestion generated in the core cloud servers. The computation offloading to vehicular edge computing, instead of the cloud, results in energy conservation in the mobile devices and reduction of networking congestion.

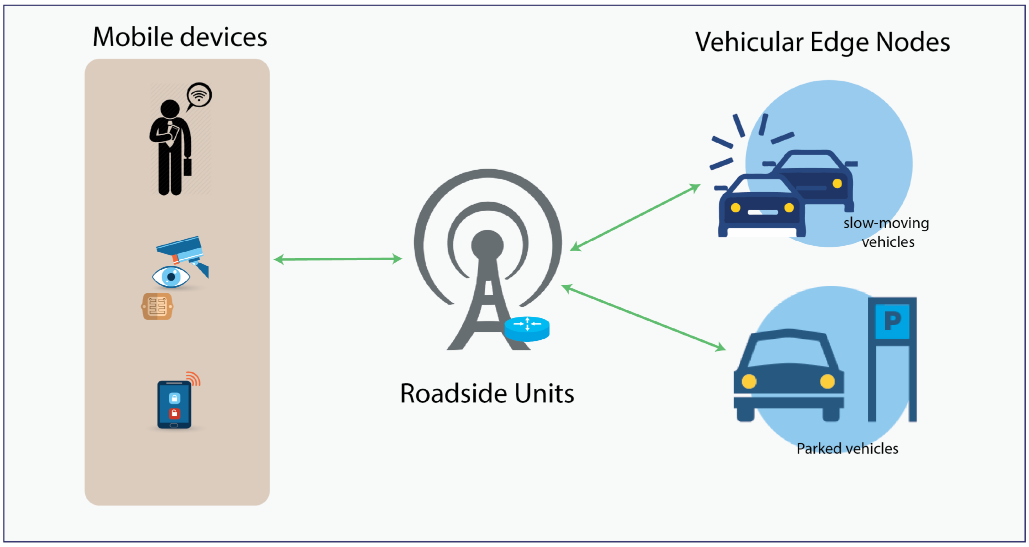

Fig. 3. General architecture for computation offloading in vehicular networks.

One typical scenario where VEC might play an important role is to assist 5G applications [Bao et al. 2018; Guo et al. 2018; Rony et al. 2018]. 5G networks will provide high-speed network coverage for mobile devices and Internet of Things (IoT). In this regard, it will support the delivery of massive volume of data from IoT devices, which would be processed in powerful servers. However, multimedia streams will compete for the same network resources. VEC can provide computation resources on the edge for IoT and 5G mobile applications. Hence, data and applications from resource-constrained smart mobile devices can be processed on vehicular edge nodes and then the result returned after the computation.

3.1.1 Preliminary Works. The works discussed in this section were proposed in the literature to address different aspects for enabling edge computing in vehicular networks. In an attempt to alleviate the load in cellular networks, Yuan et al. [2016] presents a mathematical framework to study time and space constrained data in a Vehicular Ad-hoc Network (VANET) scheme. Considering vehicular clouds are relatively new, there is still a number of challenges to address. The mathematical framework presented here contemplates the probability of contact between vehicles, so they can share content. Even if several users subscribe to the same content, they have different delivery deadlines, which makes such a solution possible. The analysis and experiments prove this framework to perform well and achieve high offloading efficiency. Al-Rashed et al. [2017] studies different classical assignment policies together with their performance and task failure, the policies studied were: best fit, first fit, and last fit. The analysis shows that last fit minimizes the total task failures, and best fit achieves a better task assignment failure rate. A cloud offloading framework for VANET, which aims to enhance the performance of this technique by using different scheduling policies to adapt for interactive applications is presented in Ashok et al. [2016]. Three policies are used for the study and each aim for a different objective: minimize CPU expense, minimize network expense and minimize the maximum lateness of applications. This solution for offloading interactive applications is tested using a prototype of a cloud system, it shows to efficiently adapt scheduling while applications meet their deadlines.

Concerning location-aware content is presented in Malandrino et al. [2017], it focuses on caching architectures for highly localized content and the impact that growing traffic demand has on them. This work uses an app, which collects location and connectivity information from the user. The assumptions of the architecture are as follows: The app can be installed on a mobile device, the user can be a static or moving pedestrian, either walking or inside a vehicle, and the content is categorized in types. The analysis presented shows that the cache sizes are influenced by the architecture used, and the cost of moving from a centralized to a distributed architecture is not significant. This study shows that caching architectures can be adapted to the edge computing scheme. However, a more thorough analysis of different communication technologies is essential to have a more clear overview. Working with delay-tolerant data, a system with offloading spots for data storage is proposed in Baron et al. [2017], it uses strategic locations for vehicles to offload massive delay-tolerant data to obtain a scalable and centralized offloading architecture.

Furthermore, in the context of VEC, a study [Refaat et al. 2016] of virtual machine (VM) migration, together with its challenges in performance. This work presents an algorithm for VM migration for vehicular clouds, which detects an exiting vehicle from the grid. The grid refers to vehicles in communication with an RSU, and exiting vehicle means a vehicle that is leaving this grid. To avoid data loss, it must be retained within the grid. Therefore, if an exiting vehicle is not able to find a destination vehicle for the data migration then it is stored in the RSU. The authors propose three different modifications to this algorithm: start looking for a destination vehicle at street level, select the destination vehicle based on the workload of the nodes, and select the destination vehicle based on the time they have left in the grid. While the idea presented is promising, the analysis showed that the results are still not satisfactory for an efficient system.

3.1.2 Challenges for VEC. Despite the significant advances encountered in the literature, there still are several challenges to be addressed towards the full development of vehicular edge computing. The primary challenge in VEC is communication between nodes for task offloading and retrieval, due to the fact that mobility is highly present and topologies are constantly changing [Nikhat and Mehmet-Ali 2018]. Considering there are no standard parameters yet to evaluate performance in vehicular edge computing, using vehicles as infrastructure show a great potential to boost the performance of computation and communication for an edge environment [Hou et al. 2016]. Wang et al. [2016] shows there is a direct relation between the delay tolerance, connectivity, the mobility of the vehicles, and the serviceability of VEC. There are other parameters to consider, such as geographical location and security, but overall VEC can be expected to enhance the performance of cloud systems.

4 TAXONOMY

There are some works in the literature that proposed a taxonomy for classification of computation offloading approaches:

- Akherfi et al. [2018] defines the process of computation offloading with the following steps:
  - *Application Partitioning*, where the application is divided into offloadable and non-offloadable tasks.
  - *Preparation*, to arrange everything that is needed for the partitioned tasks to be offloaded, including server selection.
  - *Offloading Decision*, which is the final step of the process and it refers to sending the tasks to the remote server.
- Orsini et al. [2015], considers a design guideline for the offloading system as follows:
  - *Partitioning or Pre-Phase to Offloading*, to define and divide what parts of the application need to be synchronized after the offloading process finishes.
  - *Offloading*, referred as the most important step of the process, it is where the tasks are sent to a remote server.

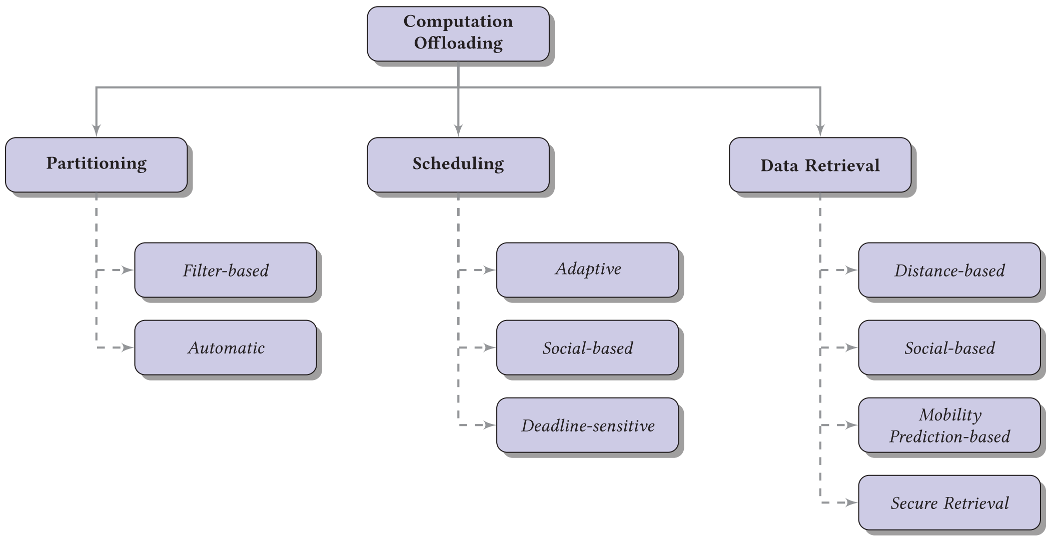

Fig. 4. Taxonomy of the process of computation offloading.

—*Handling of a Global State*, where the system stays aware of the mobile nodes and considers the rapid change of the environment.

Although these categorizations consider the essential steps for computation offloading, if the system used is the one of Vehicular Edge Computing there are some steps that have a significant influence on the performance of the technique, such as the retrieval of the output data. Therefore, we propose a taxonomy, represented in Figure 4, to classify the proposed solutions for computation offloading, as:

- *Partitioning*: Partition techniques are divided in two subcategories based on their objectives:
  —*Filter-based*, where the algorithm aims to enhance performance regarding a single parameter like energy consumption or delay;
  —*Automatic*, where the objective of the technique is to enhance overall performance in the designated environment, which in this case is edge computing.
  The works surveyed in the above subcategories present solutions for delay challenges, such as reducing complexity to lower execution time.
- *Scheduling*: Scheduling mechanisms are classified into three subcategories:
  —*Adaptive*, where the technique aims to adjust to the opportunistic nature of a mobile network to achieve an optimum performance, considering the high-mobility of nodes in VEC and the constant changes in the network's topology, it is essential to have a technique that can adapt to these changes;
  —*Social-based*, which take advantage of the mobility of the nodes and the frequent contacts of the users to reliably allocate resources according to their physical and social distances;
  —*Delay-sensitive*, which refers to the algorithms that have the only purpose of enhancing the performance of the system, regarding deadlines and allocation delay.
- *Data Retrieval*: Data retrieval methods are divided into three subcategories. Mobility is one of the main challenges in VEC considering the high mobility of the nodes. Solutions for data retrieval can be classified into four categories:

- *Distance-based*, where the geographical location of the nodes is the main parameter considered for a successful retrieval of the output data;
- *Social-based*, which with the same principle of the distance-based uses social relationships to obtain a more secure and efficient data retrieval;
- *Mobility prediction-based*, which refers to techniques that take advantage of the mobility of nodes to predict the location of the nodes to calculate the best path for the user to retrieve the data packets.
- *Secure retrieval*, which refers to protocols that consider and propose security mechanism for dealing with privacy and attack concerns during data retrieval from vehicular nodes.

The novelty of the proposed classification is that a category for data retrieval is included to address the mobility of the vehicles processing the tasks. Moreover, both the scheduling and the data retrieval include a social-based subcategory which consider interaction of the edge nodes to achieve an efficient offloading and retrieval. Furthermore, the steps are classified into subcategories based on the existing works and future directions for vehicular edge computing.

5 PARTITIONING

The rapid development of technology in mobile devices has brought an increasing popularity of developing compute-intensive applications. Even when portable devices, such as mobile phones, tablets, laptops or even watches have new and more powerful computing resources every year, sometimes these are not enough for high-performance applications such as gaming, live-streaming or virtual reality applications. To support all of these applications, Chun et al. [2011] and Cuervo et al. [2010] suggest task offloading to cloud servers. To do this, the application must be partitioned into smaller chunks of data, so there is a lower communication delay involved in the offloading process. It is important to mention that not all applications can be partitioned and offloaded to be executed in cloud servers. Moreover, task partitioning and offloading procedures can take into consideration several user and network parameters (e.g., cost, energy, bandwidth, and delay) for deciding whether a task will be partitioned and which parts will be offloaded to be executed in a cloud server.

For applications to be divided into smaller tasks, first those tasks have to be classified into classes. These classes can be defined based on many criteria, for example, the importance of the task in the application. In the case of computation offloading in a mobile environment, such as vehicular edge computing, a major parameter to consider is whether the task can be executed externally. Some applications may need information from sensors, such as GPS or the camera, that can only be obtained from the mobile device. Once these parameters are defined, and the application partitioned into smaller tasks, it is much simpler to offload them to be executed in an external server or device. Figure 5 represents the topology used in Niu et al. [2017], which is a typical partitioning topology for a mobile device application.

Being the first step of the process, it is essential for the partitioning algorithm used to analyze the environment before starting to define parameters. In the case of vehicular edge computing, the most important characteristic to evaluate is the mobility of the nodes and the possibility of connectivity failure. Considering these two constraints, it is logic that a partitioning algorithm that breaks compute-intensive tasks into smaller tasks can help the nodes to reduce the execution delay and make the following steps more efficient. For scheduling, having an efficient partitioning algorithm means a more seamless resource allocation and task execution. Furthermore, for data retrieval, by reducing the overall delay implies that the mobile node will have fewer drawbacks considering that the distance traveled while processing the task is short.

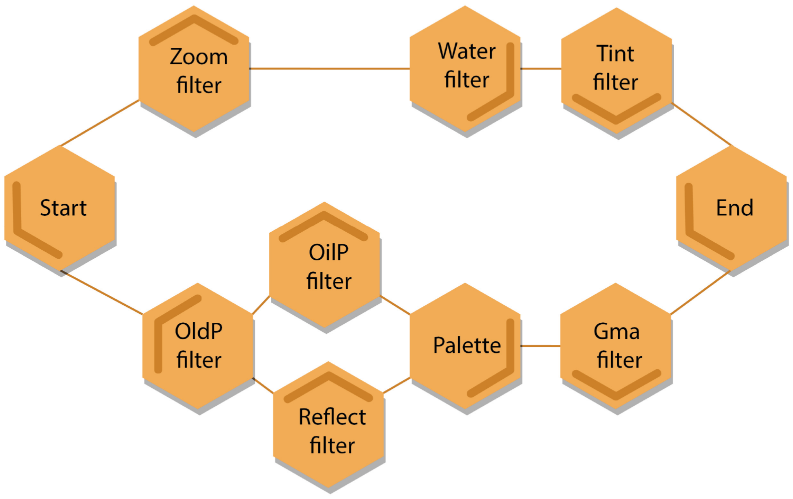

Fig. 5. Topology structure of an eight-node application.

Table 1. Partitioning Approaches for Computation Offloading

Approach	**Advantages**	**Disadvantages**
Filter-based	—Identifies the more convenient tasks to offload, based on a specific preset parameter	—Does not consider any parameters outside of the preset ones —May not obtain the best performance
Automatic	—Identifies tasks to offload based on the available resources	—May consume additional resources to verify availability

Partitioning can be categorized in different ways [Liu et al. 2015], although there are no standard parameters to evaluate vehicular edge computing. The works reviewed in the proposed subcategories are considered to be potential techniques to perform well if used for computation offloading with mobile nodes. The categorization of the existing work in partitioning for computation offloading is summarized in Table 1. Furthermore, the discussed solutions in each category are qualitatively compared in Table 2.

5.1 Filter-Based Partitioning

Filter-based partitioning in the sense of computation offloading refers to the process of identifying the more convenient tasks to offload. Different parameters can be considered for filter-based partitioning, but in general, the reduction of resource consumption is the main objective. The process for a filter-based partitioning begins with defining what is the objective of the technique. For example, if the more complex tasks are going to be offloaded, then partition the application accordingly. The advantages of this approach are that the algorithms can be adapted for any specific constraint in the scheme it is used. However, because of the same reason the drawback can present when the environment considered has a few open issues and it becomes problematic for the partition algorithm to achieve an optimal performance. This method becomes practical for applications, such as language translation and voice recognition. There are several techniques that can serve for filter-based application partitioning [Catalyurek et al. 2007; Karypis and Kumar 1999; Khandekar et al. 2009]. However, most of them are designed for wired networks and may not perform as well in a wireless scheme, which is the one considered in this study. Because the amount of work

Table 2. Representative Algorithms for Partitioning in Computation Offloading

Proposal	Category	Objective	Method	Potential Application
Elicit [Hassan et al. 2015]	Filter-based	Reduce response time & energy consumption	Identify the most compute-intensive tasks	Mobile edge computing (Language translators)
[Niu et al. 2017]	Filter-based	Reduce computation costs	Exploit user preferences to define a more personalized partition plan	Mobile applications (multimedia, gaming)
[Jungum et al. 2016]	Filter-based	Reduce network traffic	Use graph theory and clustering to enhance the usage of memory and CPU	Pervasive computing
[Ajwani et al. 2016]	Filter-based	Minimize partitioning cost	Use graph theory together with an architecture-oblivious partitioning	Long-duration streaming applications
ParGen [Wen et al. 2017]	Automatic	Obtain optimal throughput	Enhance overall performance based on a genetic algorithm	Data streaming
[Calice et al. 2015]	Automatic	Reduce computation time & energy consumption	Split tasks based on the available resources	Augmented reality
[Zhang et al. 2018b]	Automatic	Reduce energy consumption	Cluster offloadable and non-offloadable tasks to obtain the optimal partitioning solution	Speech recognition
[Niu et al. 2014]	Automatic	Minimize execution cost	Use a weighted object relation graph to model the applications before partitioning	Image detection

related to edge computing is limited, the following works focus on different task profiling and are considered to potentially enhance the performance in vehicular networks with some adaptations.

Elicit [Hassan et al. 2015] is a framework proposed to efficiently identify the compute-intensive tasks in mobile applications for offloading. This framework considers not only the profile of the mobile application but the environmental setup. The bandwidth between the server and the mobile device, network latency, and size over the overhead data are contemplated to make the optimal partition of the task based on the results obtained. The purpose of Elicit is to improve the execution time of the tasks. To do this, they partition the mobile application and offload the most resource-consuming ones, which are assumed to be known already. To select which method is to be offloaded, they first divide the processes into classifications. One of these is the Constrained Class where they have the methods that cannot be offloaded. After the separation process, the algorithm filters the methods by examining some parameters like execution cost in the mobile device, execution time on the server, and the class of the process. These filters are there to check none of the tasks are constrained to be offloaded. For the transmitting mechanism, they follow the same principles as the POMAC transparent mechanism proposed in Hassan et al. [2014], which can offload methods that do not access any global or class variables. In addition to the method input parameters, Elicit can offload processes that access global or class variables. After the evaluation, the results show that Elicit can work with existing mobile applications and find the best method to offload the most compute-intensive part to the cloud or nearby servers via edge computing and obtain a reduced execution time and energy consumption.

An approach for reducing network traffic is proposed in Jungum et al. [2016]. The algorithm proposed models the applications as a graph with classes to then group these classes into clusters for a partition solution. The solution proposed in this work is categorized as a filter-based technique due to the fact that it has the main purpose of minimizing network traffic to enhance performance of pervasive computing schemes. The partitioning process starts with modeling the applications using graph theory, which models them as undirected graphs with a set of vertices, that represents the class or granularity of the applications, and a set of edges representing interaction between classes. After the application is modeled the clustering algorithm proposed aims to achieve a balanced load and memory usage for the CPU, therefore obtaining an efficient usage of resources and network traffic. In order to evaluate the performance of the proposed solution, experiments are carried out using a prototype. Although offloading to a remote server is not evaluated in these experiments, the partitioning phase shows to improve CPU and memory usage which should lead to the minimization of network traffic.

A similar work [Ajwani et al. 2016] proposes a system to optimize the performance of a network with long-duration streaming applications. In the presented scheme, a partition algorithm is developed, which models the applications as a graph and it adapts to the current topology of the system to achieve the optimal distribution of tasks. This partitioning mechanism aims to minimize the partitioning complexity and to balance the load in a changing topology.

Regarding filter-based techniques in terms of user preferences, Niu et al. [2017] presents a partition algorithm that aims to achieve an efficient personalized division of tasks for users. The objective of the proposed algorithm is to get the optimal performance in partitioning and offloading tasks regarding execution costs. The proposed approach makes use of a user profile model, which includes parameters like how often the mobile device is charged and memory cleaning frequency. These parameters are used as a training set to build the execution cost model. The cost model calculates the average delay and energy consumption for processing a task. In addition, all the information gathered by these models is then used by the max-cut partitioning algorithm, which finds the optimal partition plan. In order to assess the performance of the partition algorithm, the authors compare it to some other techniques and it shows to outperform them. Although the algorithms used for comparison are not based on user preferences, the behavior of the user is an essential parameter to consider when the main objective of the proposed solutions is to achieve the best QoS.

5.1.1 Discussion. One of the critical challenges in filter-based task partition is the proper consideration of the parameters to identify which tasks should be offloaded. Many parameters are conflicting and they should be taken into consideration during task partition and offloading decisions. For instance, a given task might be computation and energy-intensive (e.g., voice recognition and voice commands identification) but also might have hard-deadline constraints. Therefore, from the application point of view the task might not be offloadable as the system might not guarantee to process and deliver the result within the delay requirements, given network congestions and cloud/edge computation demands.

It is always essential to understand the application before selecting the partitioning algorithm. Considering that, in vehicular edge computing, some of the tasks can be executed in vehicles, it is important to recognize that the less complex the task is, the lower is the less execution delay. The technique proposed in Niu et al. [2017] is based on a profile created for every user. In the context of vehicular edge computing, this approach seems more appropriate for the environment. The parameters considered for the profile are related to the behavior of the user. Therefore, each device will have different classes of tasks to offload and an adaptable QoS can be provided for the users. Even though the studies in Ajwani et al. [2016] and Jungum et al. [2016] do not perform experiments specifically on computation offloading, the partition solution aiming to enhance the performance in memory and CPU usage makes sense to lower the network traffic. These parameters are essential for a partitioning scheme to be considered in an overloaded environment, such as edge computing.

Machine learning (ML)-based techniques can play important in such multi-parameters task partitioning and offloading. For instance, ML can be used for determine users profile [Furletti et al. 2012; Karaliopoulos et al. 2016] to predict resource usage of their resource-constrained smart mobile devices (e.g., energy consumption and battery recharge habits). Such approaches can assist filter-based task offloading by considering predicted values of important parameters rather than the only observed current status, during task offloading decisions.

5.2 Automatic Partitioning

In the context of computation offloading, automatic partitioning refers to identifying the tasks that can be offloaded to achieve an efficient task offloading based on the available resources at runtime. Unlike the filter-based technique, this method focuses on enhancing performance based on more general parameters, such as execution cost and optimal throughput. The advantage of automatic partitioning is that the algorithm considers the available resources, which is useful in a scenario with high mobility nodes. However, this technique aims to enhance the overall performance and cannot focus on single constraints. It is good for applications that need a rapid response, such as augmented reality or data streaming. There is little literature focused on this type of partitioning. However, representative work is discussed here.

Calice et al. [2015] presented a distributed architecture for mobile-to-mobile offloading with an automatic splitting and scheduling algorithm. The scenario presented is with an offloader, referring to the mobile device that wants to push compute-intensive applications to another mobile device, and the neighboring offloadee nodes, which refers to the mobile devices that are in range at a time t. The splitting/scheduling algorithm designed and implemented in the experiments estimates the time for a task being run locally, and the required time to offload the task to an offloadee. Each device keeps a history of the tasks that have been executed locally, together with time attributes for each one in a vector. This vector is exchanged and updated every time a new offloadee device is found. Additionally, it is utilized in a linear equation where they consider data transmission delay, execution time, bandwidth and CPU speed to get the optimum splitting/scheduling for each task. After conducting analysis and experiments using real applications and data, their results show a 99.7% of efficiency for the splitting and offloading part and up to an 80% of energy saved when offloading a task to be executed.

Regarding task partitioning for multi-user scenarios, Wen et al. [2017] proposes ParGen, a computation partitioning scheme for data streaming applications, such as augmented reality. These applications use the camera on the mobile devices to collect images as an input and after analyzing them, an output is presented. The partitioning approach proposed is based on a building recognition application for students, where new students point their cameras at buildings and the name of the building and some relevant information is presented on their screens. The ParGen method, based on a genetic algorithm, separates users requesting the data into groups. Next, it adjusts the bandwidth of the server according to the groups. Finally, the best partition and allocation path is selected, based on average overhead. The proposed solution in this work is compared against other genetic algorithms and shows to outperform them. However, the simulation environment used for the experiments assumes that all the mobile devices are executing the same application, hence the same amount of data is being requested. In a MEC environment is important to consider that a significant part of the nodes have mobility, and it is not realistic to assume all of them will request the same data.

A bandwidth-adaptive partitioning scheme is presented in Niu et al. [2014]. It aims to find an optimal partitioning solution for small applications and to reduce execution cost. After a model of the application is built using a weighted object relation graph, the partitioning algorithms, Branch-and-Bound-based Application Partitioning (BBAP) and Min-Cut-based Greedy Application Partitioning (MCGAP) algorithms, check the bandwidth changes and generates the partition models to minimize the execution costs. In order to evaluate the performance of the algorithms, simulations were performed using both of them in different scenarios. While BBAP shows to perform well with small applications, the MCGAP algorithm shows to be fit for larger ones and both of them achieve reduction of the execution cost.

Zhang et al. [2018b] propose an algorithm for application partitioning that focuses on multi-site distribution. Although it aims for energy efficiency in a mobile cloud computing scenario, it can be considered as an automatic partitioning scheme due to the fact that it considers different locations and adapts for execution in heterogeneous devices. The process adopted for application partitioned in this work follows the next steps: application modeling, cost modeling, and partitioning solving. An energy model is used to achieve the optimum energy consumption. This model uses a data cost graph, which involves data from the CPU workload, the bandwidth rate and performance of the different locations considered for offloading. The authors propose a cyclic random movement genetic algorithm (CRMGA) to find the optimal partitioning solution for mobile applications. This algorithm first clusters all the unoffloadable tasks and assigns them to the local storage and then finds the optimal solution to partition the offloadable tasks. The experiments to evaluate this scheme are carried out on a prototype and the parameters used are based on energy consumption and the time spent obtaining the partitioning solutions.

The proposed solutions for automatic partitioning reviewed here to consider whether it is convenient to execute the tasks on a remote server. The first work discussed [Calice et al. 2015] keeps a history of the tasks that have been offloaded and the ones executed locally. Furthermore, the approach in Wen et al. [2017] creates clusters of nodes based on the requests received and adjusts them depending on the available bandwidth. Moreover, the work in Niu et al. [2014] also adapts to the changes in bandwidth and can be used in a vehicular edge computing as it is designed for small applications, which translate into low execution time. Although both are different solutions for the same problem, the two seem to achieve the expected performance. In the context of vehicular edge computing, both of the techniques discussed show to be useful for nodes with mobility. Although the proposed algorithm in Zhang et al. [2018b] is directed for mobile cloud computing, it considers mobile applications and the possibility of offloading them to different locations with heterogeneous devices. Based on these characteristics, this last work is categorized as automated partitioning and we consider it is a solution that can help further enhance the performance of edge computing if adapted adequately. Creating clusters according to the requests is favorable because different nodes are active at different times. On the other hand, keeping track of where the tasks were executed and their performance can significantly improve forthcoming application requests.

5.2.1 Discussion. As already mentioned, automatic partitioning considers more general user-side and network-side parameters (e.g., cost and optimal throughput) for partitioning and offloading decisions. One of the critical challenges will be the efficient parameters exchange among mobile users and edge devices for task partitioning and offloading decisions. For instance, the proposal in Niu et al. [2014] makes partitioning and offloading decisions based on changes in the network bandwidth status. Hence, mobile nodes should keep monitoring this aspects, which will incur in extra energy consumption and network overhead. This aspect is even more challenging for the

Table 3. Scheduling Approaches for Computation Offloading

Approach	Advantages	Disadvantages
Adaptive Scheduling	—Can adjust to changes, such as the frequent search of resources in a mobile scenario	—Complexity may generate execution delay —Difficult to guarantee an optimum performance
Social-based	—Exploits close contacts, therefore reliable nodes, for a balanced task offloading. —Considers priority for tasks.	—Security challenges —May be considered invasive by users.
Deadline-sensitive	—Aims to decrease the execution delay of the task —Can help to decrease communication cost in retrieval	—May not be the best option to consider for a high number of task requests

considered scenario of vehicular edge computing. This is because available resources on the edge will change frequently due to vehicular mobility.

6 SCHEDULING

The task offloading technique is suitable for enabling mobile users to offload their compute-intensive tasks to a remote machine. However, because the number of mobile devices and users in this technology, the network is susceptible to a bandwidth congestion, which causes an impact on the quality of service. Therefore, edge computing aims to alleviate the overload by processing tasks at the edge of the network and only sending the necessary information to the core of the cloud. Using vehicles as edge nodes can result complicated at the time of scheduling. After partitioning the application into smaller tasks, the next step is to allocate them to the edge nodes for processing. However, the nodes considered in this scenario are vehicles, and it is essential to address the mobility issues. Many scheduling policies include a decision process that, if it is not automated, the user can decide if it is convenient to offload the tasks to a remote server or to execute locally.

There are some parameters to consider for this decision process, such as the delay of communication and processing in a remote server, and sometimes is not worth it. Typically, a decision manager collects information about the resources available, resources needed, and deadline constraints for the tasks to execute seamlessly. Then, depending on the scheduling policy, the resources are allocated accordingly. In the context of vehicular edge computing, using a capable scheduling algorithm can significantly enhance the performance of computation offloading. Assuming the application is optimally partitioned, an efficient scheduling policy can guarantee a low-delay task execution. For both a high- and low-mobility scenarios, an efficient execution means that the difference of position of the nodes is not substantial. Therefore, the retrieval of the output data can be seamless and efficient as well.

Using the same example from Section 5 and assuming that the tasks have been partitioned into different classes, Figure 6 shows the model of how the partitioned tasks are sent to a decision manager, which is responsible to allocate them to a vehicle by a scheduling algorithm used in vehicular edge computing. Here, we survey some of the existing works for scheduling algorithms in computation offloading, which aim to give the user the optimum Quality of Service (QoS). Moreover, some of the discussed works are not directed for vehicular edge computing but the main idea can be adapted to the system. The categorization proposed for scheduling in computation offloading is summarized in Table 3. Furthermore, the proposed solutions for each category are qualitatively compared in Table 4.

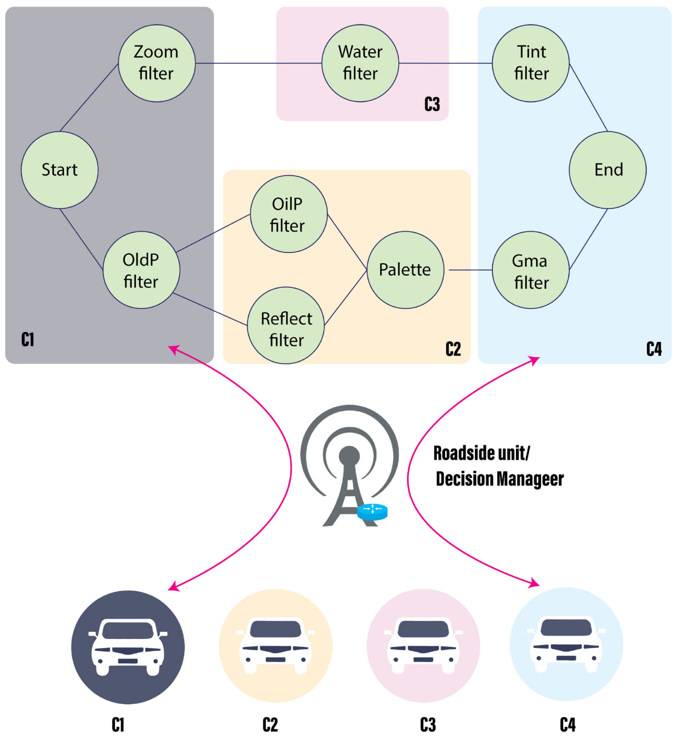

Fig. 6. General model of scheduling using a decision manager.

6.1 Adaptive Scheduling

The main motivation for adaptive scheduling is the constant search for resources caused by the dynamic nature of networks with mobile nodes. In a Mobile Edge Computing with support from vehicular infrastructure, both mobile devices and vehicles have mobility, which results in the frequent change of available resources for any region of interest considered. Using the proper scheduling policy has a significant influence on the performance of every application. Consuming time searching for available resources can translate to an increased computation delay, which means a poor QoS. The advantage of this method comes from schemes with mobile nodes, such as vehicular or mobile edge computing. However, because the main objective of adaptive scheduling is to adjust to opportunistic available resources needs to be available at all times, which can be complicated in such environment as vehicular networks. There are different approaches of using an adaptive scheduling algorithm depending on the scheme. In this section, we review some of the existing works that can be used in networks with mobility.

Midya et al. [2018] proposed a resource allocation and task scheduling policy for a vehicular cloud. The framework proposed consists of four major components: client manager, device manager, network manager, and decision engine. The client manager is in charge of the task information that arrives. The device manager controls the information about the available CPUs. The network manager collects information about the vehicular cloud and the infrastructure involved. The decision engine receives all the information collected by the managers and uses it to calculate the optimum decision about the assignment of the tasks. This approach aims to increase the number of available virtual machines and to reduce task completion time and energy consumption. It is assumed that vehicles with onboard units have access to GPS. To achieve the multiple objectives,

Table 4. Representative Algorithms for Scheduling in Computation Offloading

Proposal	**Category**	**Objective**	**Method**	**Potential Application**
HAPSO [Midya et al. 2018]	Adaptive	Scalability/Flexibility	Use particle swarm optimization to allocate a huge number of task requests	Vehicular Cloud Computing
[Mustafa et al. 2017]	Adaptive	Reduce impact of mobility in performance	Use an artificial neural network-based mobility prediction model for resource allocation	Virtual machine migration in VCC
[Chen et al. 2016]	Adaptive	Maximize overall performance (energy consumption and delay)	Use semidefinite relaxation and random mapping to allocate resources	Mobile Cloud
[Satria et al. 2017]	Adaptive	Recovery from failure	Cluster nodes with neighboring MEC cells	Real-time applications
[Yang et al. 2016]	Social-based	Reduce execution time	Load balancing by offloading tasks to friends or nodes with frequent contact	Data sharing
[Zhao et al. 2017]	Social-based	Balance throughput performance and fairness between multi-cast nodes	Create clusters based on physical and social distances to multi-cast data	Data streaming
[Wang et al. 2018]	Social-based	Lower execution delay	Achieve an optimal task assignment considering mobility of the users	Mobile applications
[Liu et al. 2016a]	Delay-sensitive	Decrease power consumption and computation delay	Analyzes the average delay and energy consumption of tasks	Mobile applications
[Zhang et al. 2016]	Delay-sensitive	Successful execution of delay-constrained tasks	Uses a contract based policy to maximize the resources available	Mobile Edge Computing
[Zhang et al. 2018a]	Delay-sensitive	Meet deadlines in an overloaded system	Model and schedule applications based on the independence of their tasks	Image/video processing
[Tang et al. 2017]	Delay-sensitive	Reduce execution time	Use an online and an offline algorithm for task assignment to provide robustness	Cloud computing

a heuristic hybrid Particle Swamp Optimization (PSO) algorithm is used. The algorithm proposed adapts with the updated information obtained from the managers before making a decision on the fittest node to achieve the lowest task completion and energy consumption. Compared to standard and self-adaptive PSO, the proposed framework shows to enhance performance based on the parameters defined.

Related to node mobility and adaptive resource allocation, the work in [Mustafa et al. 2017] aimed to use mobility prediction to avoid the negative impact of using an opportunistic network. The system used for this approach is the one of VCC, where groups of vehicles are within the communication range of the RSU to create a cloudlet. The mobility prediction model proposed is based on an artificial neural network, and the datasets used for training include the following parameters: resource lifetime, the position of vehicles, average speed of vehicles, and the number of vehicles ahead and in the same area. The performance of the prediction model was evaluated using four methods: R-squared, Pearson's correlation coefficient, mean absolute percentage error, and root mean square error. Although the mobility prediction model showed to outperform other efficient approaches, there are parameters that were not considered during the evaluation. Moreover, significant parameters, such as communication cost, workload, and network overhead, need to be considered for resource allocation to achieve an adequate performance when taken to a more realistic scenario.

Chen et al. [2016] presented a scheme to optimize the offloading decision and resource allocation for tasks in a general multi-user system where the user has several independent tasks. To do this, they propose a heuristic algorithm based on Separate Semidefinite Relaxation (SDR) [q. Luo et al. 2010]. The system considered is mobile cloud computing, where Chen et al. have one cloud server, one access point, and *N* mobile users; each one of them with a certain number of independent tasks. The algorithm proposed first solves the parameters for local execution and the settings for offloading and then compares them to see which option is more convenient. After that, it obtains the proposed offloading solution and the optimal resource allocation. The algorithm was simulated and compared to three different methods; local processing only, cloud processing only, and lower bound of optimum. The lower bound is obtained by the SDR method. After getting the results, the analysis shows that the proposed algorithm gives nearly the optimum solution to get a reduced general offloading cost (regarding energy consumption, execution time, bandwidth). However, the optimum solution is the one obtained by the same semidefinite relaxation, and while it is a good solution, it is necessary to compare it to other similar algorithms or methods and evaluate its performance in a real scenario.

Satria et al. [2017] presented a solution for an overloaded mobile edge computing environment. It proposes two schemes for recovery when there is failure to find available resources and it fits the category of adaptive scheduling due to the fact that it can adjust to new parameters to avoid complete loss of service in a MEC cell. The first of the two schemes focuses on recovery when there is only one available neighbor within range and the second scheme is directed to the cases when there is absolutely no neighboring node available for wireless transfer. The recovery algorithm finds the adjacent MEC cells, clusters them with the disconnected nodes, and chooses a cluster head to manage the recovery system to ensure it works properly. After the offline nodes are connected to new MEC cells, they can make requests through ad-hoc relay nodes, which will calculate the amount of data demanded and pass the information to the cluster head. To evaluate the capabilities of the proposed recovery mechanism in a MEC environment, simulations where performed using throughput and delay of transfer. Although the experiments were successful, it is worth mentioning that more realistic characteristics for the environment can be useful to evaluate the performance of the system in different scenarios.

6.1.1 Discussion. From the literature review, there are few existing works on scheduling for vehicular edge computing. However, due to the similarity in the architecture of the systems considered, vehicular and mobile cloud computing, it is not difficult to consider them for a vehicular edge computing scheme. One of the works reviewed [Midya et al. 2018] proposes having a manager for each of the major parts in the process, which in regarding such an opportunistic network seems like an adequate decision.

On the other hand, Chen et al. [2016] proposed to have a prediction mechanism to know the potentially available resources at all times. Choosing the adequate approach regularly depends on the application. For example, using a prediction mechanism seems suitable for an application that needs a quick response, such as video streaming or infotainment, while having a manager promises a more reliable and robust solution for applications such as virtual machine migration.

Chen et al. [2016] proposed to optimize scalability in a multi-user system, which is a mechanism that benefits the majority of applications for the environment considered in this survey. Less attention has been given on recovery of the network in situations of failure or lack of connectivity, which is a challenge that is expected in a mobile edge computing system. Moreover, the nodes used in vehicular edge computing often have high mobility, which can directly affect connectivity between nodes. Therefore, a solution as the one mentioned is essential for resource allocation and scheduling in computation offloading.

6.2 Social-Based Scheduling

Social-based scheduling techniques use data obtained from social media to develop an approach where the routine users follow day-to-day can be exploited to give them a better QoS on different applications. In VEC, the nodes that have frequent contact can be used to balance the load of task requests. The advantage of these methods is that they use data that is already available to create clusters of nodes to use as reliable resources. However, the drawback of it is that using that information from users can sometimes be considered somewhat invasive. Here we discuss existing works that use social connections to propose scheduling algorithms that aim to enhance performance for the computation offloading technique.

Yang et al. [2016] proposed a load balancing scheme based on the ball and bin theory, which is a phenomenon that can be applied to traffic balancing in distributed networks. They apply this theory in a scenario where a user needs to offload a task to a friend (mobile device contact). However, considering the distribution of the relationship between contacts is not uniform, because it is expected that the user will spend more time with some contacts than with others, and the ball and bin theory needs a uniform distribution, then using this theory will not work. Therefore, the authors proposed a new concept for balanced task offloading which they called "Your friends are more powerful than you". This new concept considers the closer social relationships between users and keeps the record of the users that have a more frequent contact. Multi-hop schemes are not considered in the proposed method, and each task is regarded as a vital issue for each user. So, if multi-hop was to be used, task completion time will significantly increase. Furthermore, tasks with a high priority are to be executed locally. The proposed social-relationship-based algorithm was compared regarding efficiency, and it shows a good performance, as well as a significant difference in execution time between strangers and friends.

In the context of the geographical location of the nodes, a resource allocation scheme is presented in Zhao et al. [2017]. This approach aims to enhance the throughput of the network by using a social-aware algorithm to fairly allocate resources. However, because the architecture considered is similar to the one of MEC, we review it in the sense of computation offloading for vehicular edge computing. The allocation algorithm proposed first groups the nodes in clusters, based on the content preferences of the users. The node that is physically closer to the other nodes is considered the cluster head, which is selected to receive the data and multicast it to the rest of the nodes in the cluster. Furthermore, to add a node to the cluster the physical and social distance between it and the cluster head are examined. After the clusters are formed, the allocation algorithm follows two steps: power allocation and channel allocation. The experiments realized in order to analyze the performance of the proposed scheme showed that using a multicast technique has advantages in terms of maintaining a good balance between throughput and fairness. Although the technique reviewed showed to perform well, the authors consider the nodes to have low mobility and to be used in an environment with vehicles, mobility is an essential parameter to consider.

Regarding user behavior in scheduling, Wang et al. [2018] proposes a task assignment solution that considers users mobility and resource distribution to achieve a reduction in task execution delay. This proposed solution examines execution time of tasks and mobility of the users between the base stations of a mobile edge computing system. Offloaded application should be executed in the amount of time that the user remains connected to a single base station or less. However, if that is not possible, then the application is partitioned and transferred wirelessly to the next base station considering the movement trajectory of the user. The algorithm presented in this work checks the queue of tasks and assigns them to the base station that will perform the execution with the lowest delay possible. In order to evaluate the performance of the task assignment mechanism, simulations were executed to measure network parameters, such as delay, task acceptance rate, and number of users. When compared to schemes that do not consider mobility of the users, the mechanism proposed shows an enhancement in performance.

6.2.1 Discussion. The major concerns with social-based scheduling is the data collection and use of sensitive information regarding mobile users (e.g., location, routines, trajectories, and preferences). Moreover, such social-based approaches do not take into consideration state of the network and resources availability. Mostly, they are limited to use frequent contacts to select them as secure nodes and reliably offload tasks. However, it is significant to examine in more detail because, in both approaches [Yang et al. 2016; Zhao et al. 2017], the available resources depending on whether the other users are open to making their resources available for other devices. Considering this, the solutions proposed can turn into an unfair system and lead to unbalanced workload problems. Wang et al. [2018] studied a key characteristic in edge computing networks, mobility. Especially for an edge computing scheme in vehicular networks, it is essential to develop solutions that consider mobility of the nodes to achieve an adequate quality of service. Even though this work is focused on mobile edge computing, the task assignment algorithm proposed can easily be adapted to a vehicular edge computing system, as the base stations used in both work in a similar way.

6.3 Deadline-Sensitive Scheduling

Regarding computation offloading, delay is always a major constraint to consider. The main objective of schemes like Mobile Edge Computing is to provide a seamless experience to the user. Extra delay added to the process by not using the proper scheduling policy can also affect the communication cost when the output data is retrieved by the user. The main focus of deadline-sensitive scheduling is to reduce delay as much as possible in order to meet the deadlines set for tasks. The advantages of this methods are that by reducing delay at the time of scheduling, the overall delay is also decreased, which helps to achieve a better performance. However, because the only focus of this method is to meet deadlines, other issues, such as connectivity, may be unattended. This method is particularly useful for applications that are time-constrained, such as video streaming. In this section, we review existing works related to scheduling algorithms that aim to reduce the delay of the process.

Liu et al. [2016a] proposed the uses of task buffer, where the tasks are scheduled based on several parameters, such as the queuing state of the mentioned buffer and the processing and transmission state of the device used. As the system model considered is based on the edge computing paradigm, to take advantage of this, the applications targeted are compute-intensive and delay-sensitive. For the task queuing, the model used first divides the time into time-slots with a same predefined length, where at the beginning of each slot a task is generated. Then, the created jobs are queued in the task buffer for the mobile device to decide if the task will be scheduled and executed locally or sent to a remote server. If the decision is to execute the task locally, then, the mobile device calculates the actual processing state, to know if there is a task already being executed; to define in what time-slot the arriving task will begin. In contrast, if the decision is to send it to the MEC server, where concurrent task execution is permitted. Then, the states of transmission, power consumption and delay are calculated. To evaluate the performance of the proposed scheme, the authors simulate three scheduling policies and compare them to their proposed one. The scheduling policies used are as follows: local execution policy (all the tasks are executed locally), cloud execution policy (all the tasks are sent to a remote server), and greedy execution policy (whenever the local CPU or the remote server is idle, any task waiting in the buffer is scheduled). After obtaining the results, the analysis shows that the proposed algorithm can achieve a lower average delay in task execution.

Zhang et al. [2016] proposed an offloading scheme for cloud-based mobile edge computing in vehicular networks. The system presented focuses on the offloading of delay-constrained applications for MEC in vehicular networks. There are some assumptions considered for the scenario that is worth mentioning. First, a unidirectional road is studied, where RSUs equipped with MEC servers are deployed and individually identified. The vehicles are considered to have mobility, therefore they establish communication with the RSU that has the strongest signal at the moment. Last, only homogeneous communication is established within the network. Considering there are many types of vehicles based on their resource capacity, a contract-based scheduling policy is developed. The RSUs in the network broadcast information about the contracts, which the vehicles can obtain. Based on the information collected, the nodes process the utility for offloading a task. If the contract brings negative utilities, such as computation delay, then it is not convenient for the vehicle to send the task and it is processed locally instead. The offloading system aims to satisfy the task requirements and exploit the utility of the MEC scheme. Even though the analysis shows that the proposed approach maximizes the use of resources, an important parameter to consider in vehicular networks is scalability. It is essential to analyze the scenario with a high traffic density of nodes and study the effect this has on bandwidth consumption because of the contract information being broadcast.

Zhang et al. [2018a] proposed a resource allocation and task scheduling scheme that aims to optimize the performance of mobile cloud systems. However, the logic of the mechanism presented can be adapted to vehicular edge computing. The load-aware resource allocation and task scheduling (LARTS) algorithm is meant to give its best performance when the cloud is overloaded. In this situation, all the tasks are distributed again so the delay-tolerant tasks are given a lower priority than the deadline-sensitive ones. The algorithm uses directed acyclic graphs to model applications and takes advantage of the differences and independence of both delay-tolerant and delay-sensitive tasks to do the load-balancing and scheduling based on a genetic algorithm. In order to evaluate the performance of the algorithm proposed, the authors realize experiments using parameters, such as arrival rate and cost of migration. The mechanism proposed shows to enhance the overall performance of the cloudlet-based system which proves that using it on an edge computing environment, which has a similar architecture, can potentially improve the quality of service.

A load balancing scheme is presented in Tang et al. [2017], it aims to finish tasks as fast as possible in cloud computing systems. The authors use online and offline methods in order to achieve the optimal scheduling solution. The authors use online and offline methods to to the task assignment in order to achieve the optimal scheduling solution. The online algorithm is randomized and follows a Poisson distribution. The offline mechanism is based on a bacterial foraging optimization algorithm, which provides high robustness and high speed. Therefore, it potentially reduces delay overall. In order to evaluate the performance of both algorithms simulations are performed using heterogeneous nodes. The two algorithms show to be efficient, the online algorithm dealing with the task assignment and the offline one providing stability on the system.

6.3.1 Discussion. Deadline-sensitive scheduling in VEC is challenging due to spatio-temporal characteristics, as well as mobility of vehicles. The vehicular density changes based on spatio-temporal aspects, i.e., geographical area (downtown or suburb) and hush hours. This will affect the resource availability for mobile applications. Hence, during task scheduling, such aspects should

Table 5. Categorization for Data Retrieval Techniques in Computation Offloading

Approach	Advantages	Disadvantages
Distance-based	—Uses location based on GPS from OBU and RSU to accurately forward data packets	—Beacons to obtain location add delay. —Obtained location needs to be precise. —May generate message redundancy leading to broadcast storm.
Social-based	—Uses contact between nodes to categorize them into clusters and use nodes with frequent contact as reliable forwarders	—Relies on neighboring nodes to forward data. —May be considered invasive. —Not good for low traffic densities.
Mobility Prediction-based	—Using traffic prediction can lead to a better use of resources, therefore a more efficient retrieval of data.	—Accuracy of prediction algorithms may not always be accurate, leading to data loss.
Secure Retrieval	—Protect users privacy and security by encrypting information	—Failure of system may lead to information leaks

be taken into consideration as it might be more effective to schedule a task execution in the cloud rather than in a VEC architecture when the observed area is presenting low vehicular density. In this case, the use of a VEC will affect the delay-sensitive application performance as the task might need to wait in a queue for available resources and/or suffer additional delays due to disruptions in the vehicular network connectivity.

7 RETRIEVAL

Computation offloading, previous to mobile devices, did not have to deal with connectivity issues or communication delay due to the mobility of the nodes. However, the rapid development of communication capabilities of vehicles and mobile devices makes data dissemination an essential topic of research for the development of applications in vehicular networks.

The retrieval process in computation offloading starts when the offloaded tasks are processed and the output data needs to be collected by the user. Depending on the approach taken for the offloading, the node responsible for the output data can be an RSU or a vehicle. The main challenges for data retrieval presented in an environment where nodes have high mobility are intermittent connectivity and broadcast storm [Coutinho et al. 2018]. There is literature that proposes dissemination techniques for vehicular applications. Nevertheless, works in the literature focus on dissemination as a way of diffusion and not for retrieval as it is needed in computation offloading. In the remainder of this section data retrieval techniques for computation offloading are surveyed.

Although a considerable amount of research is directed to dissemination techniques for applications in vehicular networks, such as content delivery, emergency notification, traffic control, and intelligent transportation systems, the area of retrieval in computation offloading requires more study. The majority of the literature is focused on data dissemination as a mean to spread messages. However, computation offloading requires data forwarding in the sense that a node retrieves the output data of a task that was offloaded to an external server. There are many existing dissemination techniques that can be adapted to CO and, as mentioned before, most of them are classified into push, pull, hybrid. Therefore, here we discuss existing techniques that can potentially be adapted to data retrieval for computation offloading with vehicular infrastructure as support. The categorization proposed for data retrieval in computation offloading for vehicular edge is summarized in Table 5. Furthermore, the proposed solutions for each category are qualitatively compared in Table 6.

Table 6. Existing Algorithms for Data Retrieval

Proposal	Category	Objective	Method	Potential Application
[Soto et al. 2017]	Distance-based	Alleviate MEC network bandwidth using vehicles as support infrastructure	Uses a timer based on the distance between node and destination to avoid message redundancy and achieve a low communication delay.	Computation offloading for mobile applications
[Chang et al. 2016]	Distance-based	Energy consumption,	Use power adjusting technique to reduce energy consumption	Emergency Notifications
[Oliveira et al. 2017]	Distance-based	Reliability on dissemination	Use distance between nodes to select the optimal relay node	Safety message broadcasting
[Farooq et al. 2017]	Distance-based	Interference, packet delay	Use delay-tolerance of applications to ensure every message reaches its destination on time	massive IoT networks
[Li and Li 2013]	Social-based	Improve service availability/data forwarding performance	Introduces social distance as a parameter, the forwarding algorithm finds nodes with the lowest distance to use as a forwarder.	Computation offloading for mobile applications
[Liu et al. 2017b]	Social-based	Communication cost	Uses a low-complexity algorithm to reduce overall cost	P2P, mobile social networks
[Liu et al. 2017a]	Mobility prediction	Dissemination delay	Uses public transport as a gateway for message dissemination	Safety message distribution
[Malandrino et al. 2012]	Mobility prediction	Alleviate cellular network	Uses mobility predictions from a traffic manager to select forwarding nodes.	ITS content download
[Liu et al. 2016b]	Secure Retrieval	Fast decryption and low storage overhead	Proposes a security policy which delegates encryption/decryption to the nearest RSU	Emergency messages
[Mondal and Mitra 2016]	Secure Retrieval	Scalability in the network	Uses time stamp defined message authentication to achieve a more secure dissemination	Message dissemination
[Mondal and Mitra 2017]	Secure Retrieval	Reduce network and communication overhead	Revokes nodes based on their behavior	Message dissemination

7.1 Distance-Based Data Retrieval

Considering the main challenge in the scenario of computation offloading for MEC in vehicular networks is mobility [Diniz et al. 2017], it makes sense to use location as the key parameter of the retrieval. Assuming GPS is available for the vehicles and the RSUs in the network, which is feasible, using location can help the retrieval method to forward data packets to the accurate location. Hence, achieving an efficient retrieval with low communication delay. The advantages of this method are that by using distance as a parameter, the probability of a broadcast storm due to message redundancy can be reduced by sending the data packets to a particular location [Boukerche and Coutinho 2019b; Yu et al. 2017]. However, the drawbacks are that broadcasting the data can lead to security issues. Although there is a lack of research literature in this area, a representative work is reviewed here. It is essential to mention that some of the surveyed works are not directly related to data retrieval for computation offloading but, based on the technique used, can be adapted to such a system.

The REPRO protocol presented in Soto et al. [2017] uses a distance-based method for data retrieval in MEC with the support of vehicular networks. This protocol aims to address connectivity, mobility, and delay issues by using the idle cars as support infrastructure whenever the vehicle that is processing the task is out of reach of any RSU. Once an RSU is reached, using the advantage that the network topology is known, the best path for retrieval is calculated, allowing the sender RSU to retrieve the processed data with the least delay possible. Assuming the vehicles and RSUs have access to GPS and the nodes are aware of the existing topology, REPRO uses a beaconless, timer-based data forwarding technique. Once the data is ready, it is broadcast. The vehicles that receive the broadcast use the location information to set a timer, which varies in value and is greater the further away it is from the destination. This means that the node that is closer to the destination has the lowest timer. When the timer expires the data packet is broadcast by the forwarding vehicle. If the broadcast is received by a vehicle that has a greater timer, this vehicle cancels the forwarding process to avoid redundancy. This process is followed until the data reaches its destination or until the data finds an RSU, which then is responsible to forward the packet to the destination.

The advantage of using vehicles as support infrastructure in the context of data retrieval for computation offloading is that they have a significant amount of resources that are commonly unused [Boukerche and Meneguette 2017; Coutinho and Boukerche 2018; Meneguette and Boukerche 2017]. There are two scenarios that can be considered for vehicles: moving and parked vehicles. The advantages of a moving vehicle are that energy consumption is not a priority because as long as the car is running the battery is charging. However, in the case of a parked vehicle, it is important not to use all the energy. There are some research works that focus on dissemination of content by adapting the transmission power to reduce energy consumption and avoid a broadcast storm.

Chang et al. [2016] presents Adaptive Forwarding message and Cooperative Safe driving (AFCS), an efficient pull-based technique for emergency notification applications. Most of the dissemination techniques used in these type of applications rely on broadcasting the messages, which leads to a high bandwidth consumption due to the redundancy of the packets. It uses a power adjusting technique based on the distance of the nodes to save energy and at the same time reduce the possibility of a broadcast storm. Although this technique is directed to emergency notification messages, the same principle can be used for computation offloading.

A data dissemination protocol for traffic safety applications is proposed in Oliveira et al. [2017]. The scheme presented uses a multi-hop broadcast protocol, which based on the distance between nodes it adapts and uses a candidate selection algorithm to choose the appropriate node to relay the messages. The presented scheme was compared to other beaconless and beacon-based protocols and it showed an acceptable performance.

7.1.1 Discussion. The work reviewed in this section follows a logical solution, which is using the distance between the processing node and the host that needs to retrieve the output data. The advantages of this approach are two. First, message redundancy is avoided by using a timer that cancels any forwarding repetition. Second, using distance as a parameter for directing the broadcasts helps to reduce the bandwidth consumption by sending the data packets to a specific area. Finally, by using several nodes to forward the output data it is easier to recover from any issues that may present with connectivity. Moreover, it aims to achieve the lowest retrieval delay possible by using both vehicles and infrastructure, such as RSUs. However, distance-based data retrieval protocols do not cope with vehicular mobility and connectivity disruption. For instance, when a vehicle does not have any neighboring vehicle, it should employ a store-carry-and-forward approach to try to deliver the computation result when it reaches the coverage area of the client, other vehicles or RSU. This would increase the result retrieval delay and can be impractical for delay-sensitive applications.

7.2 Social-Based Data Retrieval

Social-based approaches exploit information obtained from the user, such as location, frequently visited places, frequent contacts, and interests. Considering social media and the number of sensors mobile devices have these days, this data is not complicated to obtain and it can surely assist many applications to achieve an enhancement of performance and QoS. The advantages of using social-based methods is that information about the relation of the nodes is already available and can be easily obtained from social media with permission of the users. However, sharing such information can be considered invasive by the users and refusing to share it can negatively affect the performance of the system. Because there is a lack of research work focused on data retrieval for computation offloading in general, it is important to mention that not all of the works reviewed here are directly related to computation offloading, but are to data dissemination with social-based techniques that can be adapted.

Li and Li [2013] proposed a social based mechanism for data forwarding with mobile ad hoc networks in mobile cloud computing. This mechanism is based on Distributed Hash Table (DHT), which is a technique developed for supporting a high number of nodes dynamically interacting with the network. The proposed approach aims to enhance the efficiency of data forwarding by considering users as nodes and using social relationships of these nodes to create clusters. A parameter, social distance, is introduced to represent the relationship between two users after an ID is assigned to each node. The objective of the forwarding mechanism is to find forwarders until the social distance is equal to zero, which means the data has reached its destination. The case that there are no neighbors around can be presented, in which the node sends the data packet to the closest base station and it is responsible to find a suitable forwarder. However, if there still no nodes nearby to forward the data, the process is aborted.

In the context of computation offloading for MEC in vehicular networks, low-complexity data retrieval techniques are of significant importance. Considering that the more complex an algorithm is, it will take more time to process, complexity translates to delay and communication costs in the process of CO.

There can be many approaches for dissemination techniques with low-complexity algorithms, for example, an algorithm which first establishes a multicast tree to transmit data packets with a low computation complexity, to achieve low communication costs is presented in Liu et al. [2017b]. This dissemination algorithm does not use RSUs and it takes advantage of opportunistic networks to share data. Experiments were realized using a prototype, and the analysis shows an enhanced performance as the number of nodes increases.

Although the approach proposed studies multicast in mobile opportunistic networks, the prototype used for the experiments exploits Bluetooth and WiFi connections to disseminate the packets, which are available in vehicles with on-board units. Moreover, nodes with high mobility are not studied in the literature reviewed, which is an essential parameter to consider if a dissemination technique, such as this, is considered to be adapted to a vehicular environment.

7.2.1 Discussion. Although the reviewed work was considered as a social-based data retrieval technique, the approach focuses on the selection of the nodes to forward the data and not on the retrieval rate. However, because there is a lack of research works in the area of data retrieval for computation offloading, it is essential to study all the available literature. Similar to the approaches in scheduling and partitioning, the authors in the discussed work [Li and Li 2013] use social connections between users to select a reliable forwarding node. In this case, the parameter of social distance is introduced, which is an advantage considering there are no standard parameters for vehicular edge computing. However, the drawback of depending on the availability of other users and their eagerness to provide personal information exists in this approach too.

7.3 Mobility Prediction-Based Data Retrieval

In data retrieval schemes that use vehicles as infrastructure, mobility prediction can be considered essential for the successful delivery of information. Either high or low mobility, knowing where the destination will be beforehand can save a significant amount of communication delay. Moreover, mobility prediction not only concerns the specific location of a node, but also the different levels of traffic flow existing during the day in an urban scenario. The advantages of mobility prediction are similar to the ones of distance-based techniques. However, this approach proposes to calculate the potential location of the node or the user after a determined time, which can guarantee a more accurate retrieval. On the other hand, the drawbacks of this method are that, depending on the mobility algorithm used, the complexity and the accuracy of it can lead to a decrease in the retrieval rate.

An approach for Offloading of cellular networks through ITS is presented in Malandrino et al. [2012]. The scheme presented relies on a Directed Short-Range Communications (DSRC) vehicular network to alleviate the cellular network from content delivery. It is assumed that the vehicles are able to communicate with both vehicular and cellular networks. Therefore, the users with mobile devices can take advantage of the ITS network to obtain content by downloading it directly from the RSUs or using vehicles as relay nodes to forward the data packets. It is considered that a traffic manager, part of the ITS environment, can predict the mobility of the nodes. Using this prediction, the traffic manager defines a link between two nodes that are within the maximum radio range. Two models are considered for the evaluation of the scheme proposed, RSU-driven relaying and greedy relying. The objective of the RSU-driven approach is to use the nodes contact prediction values to schedule the content delivery, whereas the greedy relying follows three steps: the downloaders broadcast a list of the content presently downloading, the relay nodes filter the requests for content that they do not have, and finally the relay decides how to deliver the available content requested.

A cloud-assisted safety message dissemination framework for VANETs is proposed in Liu et al. [2017a], which aims to reduce packet loss and message redundancy. It uses buses, which are equipped with mobile devices and VANET interfaces, as a gateway for the message dissemination. While the framework presented shows a satisfactory performance, it was only compared to flooding techniques and the simulations were performed on a single road, which needs more realistic scenarios. Even though an approach like this seems to have only advantages, there are still some open challenges to consider if a data retrieval technique for computation offloading using public transport is to be developed. Firstly, there are not many nodes in a public transport network, which, depending on the application, may not be reliable enough. Lastly, security issues might present, considering the number of public transport users it might be a vulnerability.

The work [Malandrino et al. 2012] reviewed in this section aims to offload cellular traffic through vehicular networks. Although it is not an approach for edge computing directly, the concept and architecture used are similar. Considering the lack of work in the area, it can be considered for data retrieval in a vehicular edge computing. As discussed before, there are no standard parameters to evaluate the performance of such a technique. However, the scheme presented uses a mobility prediction algorithm for the delivery of location-based content, which can significantly optimize the performance of data retrieval protocols by calculating the possible location of the user at the time of retrieval. The drawback, in this case, is relying on the mobility prediction algorithms to obtain an accurate and successful retrieval rate. Moreover, mobility prediction algorithms can sometimes be too complex and add significant delay to the process.

7.3.1 Discussion. A challenge to be tacked when using vehicles as support infrastructure in computation offloading is that vehicles are mobile nodes in an opportunistic network. It may be complicated to know what kind of computational resources are available on each of them, or at least it is something that consumes time, which is essential in offloading. The idea of using public transport as the support nodes is practical because of two reasons. The first reason is that in many cities they have schedules, which makes it more simple to predict their availability. The second reason is that, because they are very similar vehicles and many times deployed by the same company, permission issues like the ones that may present in a social-based approach are avoided and it is easier to know their resource capacity. Although the work presented is not used in a computation offloading scheme, based on the architecture used it can be adapted for it.

7.4 Secure Data Retrieval

Security in vehicular networks is an issue of significant importance in vehicular networks. The idea of intelligent transportation systems, which is to have automated vehicles on the streets, makes way for the vulnerability of hackers accessing the controls of vehicles in the network and possibly cause accidents [Mejri et al. 2014]. Besides this, there is also the theft of information by intercepting messages, such as emergency messages or shared data in computation offloading, being broadcast throughout the network. Because of this reasons, security has become a popular topic in message dissemination for vehicular networks.

For example, Liu et al. [2016b] proposes a secure message dissemination broadcasting with encryption and policy enforcement, computation cost of decryption is delegated to the nearest RSU. Policy enforcement means that messages should be enforced with an access policy and should be selectively delivered to the vehicles according to said policy. The proposed scheme provides fast decryption and reduces the overhead of storage in the vehicle.

Another scheme proposed in Mondal and Mitra [2016], uses time stamp defined Message authentication codes (TD-MAC) for data dissemination in VANET. The analysis presented studies security in data dissemination concerning the resiliency and communication overhead of the algorithm proposed. The results show that it outperforms other efficient MAC schemes.

The scheme proposed in Mondal and Mitra [2017] detects vehicles that revokes potential fake vehicle nodes to ensure the safety of data dissemination in VANETs. Although the proposed scheme shows adequate performance against the models that it is compared to, it needs more realistic scenarios for experiments.

7.4.1 Discussion. As discussed before, security is an important parameter to consider for data dissemination in vehicular networks. The works presented above propose different methods of making data sharing more secure. However, security measures still need to be adapted for different purposes, for example, aiming for a lower network overhead in the case of computation offloading. Moreover, it is essential to have experimented with realistic scenarios with the objective to build a more robust system.

8 FUTURE DIRECTIONS

This section presents a general discussion and open challenges found in the areas of partitioning, scheduling, and data retrieval for computation offloading. Moreover, there is an evident lack of research works in the area of data retrieval for computation offloading. Because of that, dissemination techniques that can potentially be used in computation offloading for vehicular edge computing are presented here.

8.1 Resource Discovery and Management

One of the critical challenges in VEC is the discovery and management of computing resources at the vehicles. In traditional vehicular cloud computing, one vehicle in the area is often selected as the header of the vehicular cloud, to collect the information of available resources in the vehicles in the neighborhood. Hence, it is responsible for assigning received tasks to vehicles and therefore to manage the overall resources of the vehicular cloud. However, VEC architectures present more dynamic scenarios, especially in terms of vehicular mobility. The traditional approach of electing one vehicle as the manager might not be suitable since it might leave the area at any moment. Hence, all the information that the vehicle collected should be transferred to the newly elected manager. This would increase the network overhead and delay for task assignment and result retrieval. An approach that can be explored is the use of distributed and redundant vehicle edge headers. The challenges, therefore, will be the design of distributed protocols for the election of a subset of vehicles as the manager of the VEC architecture.

8.2 Machine Learning-Based Task Partitioning and Assignment

Overall, task partitioning and assignment decisions to the VEC relies on the observed state of multiple users and network parameters. Current works in the literature are commonly developing a mathematical model for the optimal decision based on several criteria (e.g., energy, delay, throughput, and cost). However, such approaches are time and computing-intensive and are not suitable for highly dynamic environments, such as VEC, affected by spatio-temporal characteristics and multiple environments. In this regarding, machine learning-based solutions appear as a viable alternative for task partitioning and offloading for VEC systems. Such approaches will seamlessly make decisions based on the current observed status of the network, rather than pre-determined policies.

8.3 VEC/5G Integration

5G networks will enable novel content-intensive applications for mobile users. However, such applications might be computation-intensive and unsuitable for being processed in resource-constrained 5G mobile users. In this regard, VEC can support computation-intensive 5G applications. However, novel protocols for computation offloading and result retrieval should be proposed for 5G-enabled VEC architectures. In addition, mobility management and efficient handover should be addressed given the short-range coverage of 5G small-cells.

8.4 Mobility Modeling and Management

Mobility will play a key role in a VEC system. This is because the involved entities will be mobile. Hence, it is important to understand the spatio-temporal mobility aspects of pedestrian and vehicles. From the understanding of spatio-temporal characteristics of the considered mobile entities, more efficient protocols for task offloading and result retrieval can be proposed. For instance, an RSU can select the vehicle that will execute the task of a mobile user based on the vehicles' trajectory. Hence, vehicles that will be easily reachable at the end of the computation tasks can be selected to execute delay-sensitive tasks, while those vehicles traveling along low-density vehicular and uncovered areas might be selected to execute delay-tolerant applications.

8.5 Efficient Task Partitioning

Many mobile applications these days are developed to give the users a more automated lifestyle, therefore considering their behavior for decision-making algorithms may lead to achieving an optimal QoS and not only an overall adequate performance. The entire process of computation offloading can be a more efficient one if from the start the adequate approach is used. If the partitioning algorithm can guarantee a good performance, the scheduling algorithm can achieve better performance as well, and the same happens with the retrieval algorithm. However, it is important to perform experiments in the proper environment and make the necessary adjustments to the system.

8.6 Scheduling

The reviewed literature on a social-based load balancing scheme that considers nodes with frequent contact to be reliable regarding data sharing. This approach can become useful considering these days almost every application on mobile devices is related to social media in some way. However, they do not present experiments or analysis using real data which is necessary to see if this algorithm is functional in practical scenarios. Regarding computation offloading with the support of vehicular networks, a social-based approach can be exploited if we consider that most of us have a daily routine. The same vehicles are regularly parked at the same places, hence using resources from idle vehicles can significantly enhance the performance of the system and alleviate bandwidth consumption from the core servers.

8.7 Data Retrieval

Some of the proposed technique considers mobility on the nodes and how it impacts QoS on data forwarding for mobile cloud computing. Using social contacts as a resource to improve the performance of data retrieval, is a smart way of exploiting all the available resources to achieve a secure delivery. However, an approach of this category, which relies on other nodes to forward data, may have an impractical result for an environment with low traffic flow. Moreover, the social-based approaches reviewed here assumes that those nodes that have frequent contact can rely on each other, therefore, can be efficiently used as relay nodes. However, as with most of the social-based approaches, this can be considered slightly invasive in the sense that there is a record kept of frequent contacts and locations of users. In addition, new communication paradigms, such as the information-centric networking paradigm, must be investigated and explored for data retrieval in vehicular network scenarios [Boukerche et al. 2019].

9 CONCLUSION

Computation offloading is essential for resource-constrained devices to improve their processing and storage capabilities, as well as to provide a more reliable service. Vehicular edge computing uses vehicles at the edge of the network to offer this. Furthermore, by processing tasks at the edge of the network, it alleviates the load of the core layer cloud servers and reduces communication delay. This article surveyed existing works on computation offloading in vehicular edge computing. A categorization is proposed for each step of the offloading process: partitioning, scheduling, and data retrieval. Moreover, this article introduces the category of data retrieval in computation offloading since the edge nodes in a typical VEC architecture are vehicles, which have high mobility. Together with data retrieval techniques for computation offloading in VEC, existing data dissemination methods for vehicular applications, such as emergency messages and traffic control are considered for potential data retrieval methods in a VEC environment. Even though there is limited research in data retrieval for computation offloading in VEC, based on the knowledge obtained from the reviewed work, a possible path for future work is the development of an efficient and reliable data retrieval protocol for computation offloading in a VEC environment. However, it is reasonable to expect vehicular edge computing to be a solution to provide a more reliable service for mobile and vehicular applications in the future.

REFERENCES

N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie. 2018. Mobile edge computing: A survey. *IEEE Internet of Things Journal* 5, 1 (Feb 2018), 450–465.

Deepak Ajwani, Adam Hackett, Shoukat Ali, John P. Morrison, and Stephen Kirkland. 2016. Co-optimizing application partitioning and network topology for a reconfigurable interconnect. *J. Parallel and Distrib. Comput.* 96 (2016), 12–26. DOI: https://doi.org/10.1016/j.jpdc.2016.04.010

Khadija Akherfi, Micheal Gerndt, and Hamid Harroud. 2018. Mobile cloud computing for computation offloading: Issues and challenges. *Applied Computing and Informatics* 14, 1 (2018), 1–16. DOI: https://doi.org/10.1016/j.aci.2016.11.002

Ebrahim Al-Rashed, Mohammed Al-Rousan, and Naser Al-Ibrahim. 2017. Performance evaluation of wide-spread assignment schemes in a vehicular cloud. *Vehicular Communications* 9, Supplement C (2017), 144–153.

A. Ashok, P. Steenkiste, and F. Bai. 2016. Adaptive cloud offloading for vehicular applications. In *Proceedings of the IEEE Vehicular Networking Conference*. 1–8.

Wei Bao, Dong Yuan, Zhengjie Yang, Shen Wang, Bing Zhou, Stewart Adams, and Albert Zomaya. 2018. sFog: Seamless Fog computing environment for mobile IoT applications. In *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM'18)*. ACM, New York, 127–136. DOI: https://doi.org/10.1145/3242102.3242107

B. Baron, P. Spathis, H. Rivano, M. D. de Amorim, Y. Viniotis, and M. H. Ammar. 2017. Centrally controlled mass data offloading using vehicular traffic. *IEEE Trans. on Network and Service Management* 14, 2 (2017), 401–415.

A. Boukerche and R. W. L. Coutinho. 2019a. LoICen: A novel location-based and information-centric architecture for content distribution in vehicular networks. *Ad Hoc Networks* 93 (2019), 28–38. DOI: https://doi.org/10.1016/j.adhoc.2019.101899

A. Boukerche and R. W. L. Coutinho. 2019b. Tutorial information-centric vehicular networking: Why and wherefores, challenges, and design guidelines. In *Proceedings of the 2019 IEEE Symposium on Computers and Communications (ISCC)*. 1–2. DOI: https://doi.org/10.1109/ISCC47284.2019.8969643

A. Boukerche, R. W. L. Coutinho, and A. A. F. Loureiro. 2019. Information-centric cognitive radio networks for content distribution in smart cities. *IEEE Network* 33, 3 (May 2019), 146–151.

A. Boukerche, S. Guan, and R. E. De Grande. 2018. A task-centric mobile cloud-based system to enable energy-aware efficient offloading. *IEEE Transactions on Sustainable Computing* 3, 4 (Oct 2018), 248–261. DOI: https://doi.org/10.1109/TSUSC.2018.2836314

A. Boukerche, S. Guan, and R. E. De Grande. 2019. Sustainable offloading in mobile cloud computing: Algorithmic design and implementation. *ACM Comput. Surv.* 52, 1, Article 11 (Feb. 2019), 37 pages. DOI: https://doi.org/10.1145/3286688

A. Boukerche and R. I. Meneguette. 2017. Vehicular cloud network: A new challenge for resource management based systems. In *Proceedings of the 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*. 159–164. DOI: https://doi.org/10.1109/IWCMC.2017.7986279

A. Bozorgchenani, D. Tarchi, and G. E. Corazza. 2017. An energy and delay-efficient partial offloading technique for Fog computing architectures. In *Proceedings of the 2017 IEEE Global Communications Conference (GLOBECOM'17)*. 1–6. DOI: https://doi.org/10.1109/GLOCOM.2017.8254703

G. Calice, A. Mtibaa, R. Beraldi, and H. Alnuweiri. 2015. Mobile-to-mobile opportunistic task splitting and offloading. In *Proceedings of the 2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 565–572. DOI: https://doi.org/10.1109/WiMOB.2015.7348012

U. V. Catalyurek, E. G. Boman, K. D. Devine, D. Bozdag, R. Heaphy, and Lee Ann Riesen. 2007. Hypergraph-based dynamic load balancing for adaptive scientific computations. In *Proceedings of the 2007 IEEE International Parallel and Distributed Processing Symposium*. 1–11. DOI: https://doi.org/10.1109/IPDPS.2007.370258

B. J. Chang, Y. H. Liang, and Y. D. Huang. 2016. Efficient emergency forwarding to prevent message broadcasting storm in mobile society via vehicle-to-X communications for 5G LTE-V. In *Proceedings of the 2016 International Computer Symposium (ICS)*. 479–484.

M. H. Chen, B. Liang, and M. Dong. 2016. Joint offloading decision and resource allocation for multi-user multi-task mobile cloud. In *Proceedings of the 2016 IEEE International Conference on Communications (ICC)*. 1–6. DOI: https://doi.org/10.1109/ICC.2016.7510999

Byung-Gon Chun, Sunghwan Ihm, Petros Maniatis, Mayur Naik, and Ashwin Patti. 2011. CloneCloud: Elastic execution between mobile device and cloud. In *Proceedings of the 6th Conference on Computer Systems (EuroSys'11)*. ACM, New York, 301–314. DOI: https://doi.org/10.1145/1966445.1966473

R. W. L. Coutinho and A. Boukerche. 2018. Guidelines for the design of vehicular cloud infrastructures for connected autonomous vehicles. *IEEE Wireless Communications* (Aug 2018), 2–7.

R. W. L. Coutinho and A. Boukerche. 2020. Modeling and analysis of a shared edge caching system for connected cars and industrial IoT-based applications. *IEEE Transactions on Industrial Informatics* 16, 3 (March 2020), 2003–2012.

R. W. L. Coutinho, A. Boukerche, and X. Yu. 2018. Information-centric strategies for content delivery in intelligent vehicular networks. In *Proceedings of the 8th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet'18)*. ACM, New York, 21–26. DOI: https://doi.org/10.1145/3272036.3272048

Eduardo Cuervo, Aruna Balasubramanian, Dae-ki Cho, Alec Wolman, Stefan Saroiu, Ranveer Chandra, and Paramvir Bahl. 2010. MAUI: Making smartphones last longer with code offload. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys'10)*. ACM, New York, 49–62. DOI: https://doi.org/10.1145/1814433.1814441

G. R. Diniz, F. D. Cunha, and A. A. F. Loureiro. 2017. On the characterization of vehicular mobility. In *Proceedings of the 6th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications (DIVANet'17)*. ACM, New York, 23–29. DOI: https://doi.org/10.1145/3132340.3132349

M. J. Farooq, H. ElSawy, Q. Zhu, and M. S. Alouini. 2017. Optimizing mission critical data dissemination in massive IoT networks. In *Proceedings of the 2017 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. 1–6.

J. Feng, Z. Liu, C. Wu, and Y. Ji. 2017. AVE: Autonomous vehicular edge computing framework with ACO-based scheduling. *IEEE Transactions on Vehicular Technology* 66, 12 (Dec. 2017), 10660–10675. DOI: https://doi.org/10.1109/TVT.2017.2714704

J. Feng, Z. Liu, C. Wu, and Y. Ji. 2019. Mobile edge computing for the internet of vehicles: Offloading framework and job scheduling. *IEEE Vehicular Technology Magazine* 14, 1 (March 2019), 28–36.

B. Furletti, L. Gabrielli, C. Renso, and S. Rinzivillo. 2012. Identifying users profiles from mobile calls habits. In *Proceedings of the ACM SIGKDD International Workshop on Urban Computing (UrbComp'12)*. ACM, New York, 17–24. DOI: https://doi.org/10.1145/2346496.2346500

H. Guo, J. Liu, J. Zhang, W. Sun, and N. Kato. 2018. Mobile-edge computation offloading for ultradense IoT networks. *IEEE Internet of Things Journal* 5, 6 (Dec 2018), 4977–4988. DOI: https://doi.org/10.1109/JIOT.2018.2838584

H. Guo, J. Zhang, and J. Liu. 2019. FiWi-enhanced vehicular edge computing networks: Collaborative task offloading. *IEEE Vehicular Technology Magazine* 14, 1 (March 2019), 45–53.

Mohammed A. Hassan, Kshitiz Bhattarai, Qi Wei, and Songqing Chen. 2014. POMAC: Properly offloading mobile applications to clouds. In *Proceedings of the 6th USENIX Conference on Hot Topics in Cloud Computing (HotCloud'14)*. USENIX Association, Berkeley, CA, 7–7. http://dl.acm.org/citation.cfm?id=2696535.2696542

M. A. Hassan, Qi Wei, and Songqing Chen. 2015. Elicit: Efficiently identify computation-intensive tasks in mobile applications for offloading. In *Proceedings of the 2015 IEEE International Conference on Networking, Architecture and Storage (NAS)*. 12–22. DOI: https://doi.org/10.1109/NAS.2015.7255215

M. Hirsch, C. Mateos, and A. Zunino. 2018. Augmenting computing capabilities at the edge by jointly exploiting mobile devices: A survey. *Future Generation Computer Systems* 88 (2018), 644–662. DOI: https://doi.org/10.1016/j.future.2018.06.005

X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen. 2016. Vehicular fog computing: A viewpoint of vehicles as the infrastructures. *IEEE Transactions on Vehicular Technology* 65, 6 (2016), 3860–3873.

Nevin Vunka Jungum, Nawaz Mohamudally, and Nimal Nissanke. 2016. Partitioning application using graph theory for mobile devices in pervasive computing environments. *Procedia Computer Science* 94 (2016), 105–112. DOI: https://doi.org/10.1016/j.procs.2016.08.018 The 11th International Conference on Future Networks and Communications (FNC 2016) / The 13th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2016) / Affiliated Workshops.

M. Karaliopoulos, I. Koutsopoulos, and M. Titsias. 2016. First learn then earn: Optimizing mobile crowdsensing campaigns through data-driven user profiling. In *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'16)*. ACM, New York, 271–280. DOI: https://doi.org/10.1145/2942358.2942369

George Karypis and Vipin Kumar. 1999. Parallel multilevel series k-way partitioning scheme for irregular graphs. *SIAM Rev.* 41, 2 (1999), 278–300. DOI: https://doi.org/10.1137/S0036144598334138

Rohit Khandekar, Kirsten Hildrum, Sujay Parekh, Deepak Rajan, Joel Wolf, Kun-Lung Wu, Henrique Andrade, and Buğra Gedik. 2009. COLA: Optimizing stream processing applications via graph partitioning. In *Proceedings of the 10th ACM/IFIP/USENIX International Conference on Middleware (Middleware'09).* Springer-Verlag New York, Inc., New York, Article 16, 20 pages. http://dl.acm.org/citation.cfm?id=1656980.1657002

Z. Kuang, L. Li, J. Gao, L. Zhao, and A. Liu. 2019. Partial offloading scheduling and power allocation for mobile edge computing systems. *IEEE Internet of Things Journal* 6, 4 (Aug 2019), 6774–6785. DOI: https://doi.org/10.1109/JIOT.2019.2911455

Z. Li and M. Li. 2013. An efficient social based data forwarding mechanism for mobile cloud computing. In *Proceedings of the 2013 IEEE 9th International Conference on Mobile Ad-hoc and Sensor Networks.* 365–372. DOI: https://doi.org/10.1109/MSN.2013.59

B. Liu, D. Jia, J. Wang, K. Lu, and L. Wu. 2017a. Cloud-assisted safety message dissemination in VANET; Cellular heterogeneous wireless network. *IEEE Systems Journal* 11, 1 (March 2017), 128–139. DOI: https://doi.org/10.1109/JSYST.2015.2451156

Jieyao Liu, Ejaz Ahmed, Muhammad Shiraz, Abdullah Gani, Rajkumar Buyya, and Ahsan Qureshi. 2015. Application partitioning algorithms in mobile cloud computing: Taxonomy, review and future directions. *Journal of Network and Computer Applications* 48, Supplement C (2015), 99–117.

Juan Liu, Yuyi Mao, Jun Zhang, and Khaled B. Letaief. 2016a. Delay-optimal computation task scheduling for mobile-edge computing systems. 1451–1455 pages.

Xuejiao Liu, Yingjie Xia, Wenzhi Chen, Yang Xiang, Mohammad Mehedi Hassan, and Abdulhameed Alelaiwi. 2016b. SEMD: Secure and efficient message dissemination with policy enforcement in VANET. *J. Comput. System Sci.* 82, 8 (2016), 1316–1328. DOI: https://doi.org/10.1016/j.jcss.2016.05.006

Y. Liu, H. Wu, Y. Xia, Y. Wang, F. Li, and P. Yang. 2017b. Optimal online data dissemination for resource constrained mobile opportunistic networks. *IEEE Transactions on Vehicular Technology* 66, 6 (2017), 5301–5315.

P. Mach and Z. Becvar. 2017. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys Tutorials* 19, 3 (thirdquarter 2017), 1628–1656. DOI: https://doi.org/10.1109/COMST.2017.2682318

F. Malandrino, C. Casetti, C. F. Chiasserini, and M. Fiore. 2012. Offloading cellular networks through ITS content download. In *Proceedings of the 2012 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON).* 263–271. DOI: https://doi.org/10.1109/SECON.2012.6275786

Francesco Malandrino, Carla-Fabiana Chiasserini, and Scott Kirkpatrick. 2017. The impact of vehicular traffic demand on 5G caching architectures: A data-driven study. *Vehicular Communications* 8, Supplement C (2017), 13–20.

Toni Mastelic, Ariel Oleksiak, Holger Claussen, Ivona Brandic, Jean-Marc Pierson, and Athanasios V. Vasilakos. 2014. Cloud computing: Survey on energy efficiency. *ACM Comput. Surv.* 47, 2, Article 33 (Dec. 2014), 36 pages. DOI: https://doi.org/10.1145/2656204

V. Medina and J. M. García. 2014. A survey of migration mechanisms of virtual machines. *ACM Comput. Surv.* 46, 3, Article 30 (Jan. 2014), 33 pages. DOI: https://doi.org/10.1145/2492705

Mohamed Nidhal Mejri, Jalel Ben-Othman, and Mohamed Hamdi. 2014. Survey on VANET security challenges and possible cryptographic solutions. *Vehicular Communications* 1, 2 (2014), 53–66. DOI: https://doi.org/10.1016/j.vehcom.2014.05.001

Rodolfo I. Meneguette and Azzedine Boukerche. 2017. SERVitES: An efficient search and allocation resource protocol based on {V2V} communication for vehicular cloud. *Computer Networks* 123 (2017), 104–118.

Sadip Midya, Asmita Roy, Koushik Majumder, and Santanu Phadikar. 2018. Multi-objective optimization technique for resource allocation and task scheduling in vehicular cloud architecture: A hybrid adaptive nature inspired approach. *Journal of Network and Computer Applications* 103 (2018), 58–84. DOI: https://doi.org/10.1016/j.jnca.2017.11.016

A. Mondal and S. Mitra. 2016. TDMAC: A timestamp defined message authentication code for secure data dissemination in VANET. In *Proceedings of the 2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS).* 1–6. DOI: https://doi.org/10.1109/ANTS.2016.7947863

A. Mondal and S. Mitra. 2017. Revocation of misbehaving vehicles during data dissemination among connected vehicles in VANET. In *Proceedings of the 2017 IEEE Region 10 Symposium (TENSYMP).* 1–7. DOI: https://doi.org/10.1109/TENCONSpring.2017.8070037

A. M. Mustafa, O. M. Abubakr, O. Ahmadien, A. Ahmedin, and B. Mokhtar. 2017. Mobility prediction for efficient resources management in vehicular cloud computing. In *Proceedings of the 2017 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud).* 53–59. DOI: https://doi.org/10.1109/MobileCloud.2017.24

S. Nikhat and M. Mehmet-Ali. 2018. An analysis of user mobility in cellular networks. In *Proceedings of the 16th ACM International Symposium on Mobility Management and Wireless Access (MobiWac'18).* 74–81.

Zhaolong Ning, Jun Huang, Xiaojie Wang, Joel J. P. C. Rodrigues, and Lei Guo. 2019a. Mobile edge computing-enabled internet of vehicles: Toward energy-efficient scheduling. *IEEE Network* 33, 5 (Sep. 2019), 198–205.

Z. Ning, X. Wang, and J. Huang. 2019b. Mobile edge computing-enabled 5G vehicular networks: Toward the integration of communication and computing. *IEEE Vehicular Technology Magazine* 14, 1 (March 2019), 54–61.

Jianwei Niu, Wenfang Song, and Mohammed Atiquzzaman. 2014. Bandwidth-adaptive partitioning for distributed execution optimization of mobile applications. *Journal of Network and Computer Applications* 37 (2014), 334–347. DOI: https://doi.org/10.1016/j.jnca.2013.03.007

Jianwei Niu, Shihao Wang, Wei Niu, and Mohammed Atiquzzaman. 2017. User-aware partitioning algorithm for mobile cloud computing based on maximum graph cuts. *Computer Networks* 129 (2017), 193–206. DOI: https://doi.org/10.1016/j.comnet.2017.09.011

Renê O. C. Montez, Carlos Montez, Azzedine Boukerche, and Michelle S. Wangham. 2017. Reliable data dissemination protocol for VANET traffic safety applications. *Ad Hoc Networks* 63, Supplement C (2017), 30–44. DOI: https://doi.org/10.1016/j.adhoc.2017.05.002

Gabriel Orsini, Dirk Bade, and Winfried Lamersdorf. 2015. Context-aware computation offloading for mobile cloud computing: Requirements analysis, survey and design guideline. *Procedia Computer Science* 56 (2015), 10–17. DOI: https://doi.org/10.1016/j.procs.2015.07.169 The 10th International Conference on Future Networks and Communications (FNC 2015) / The 12th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2015) Affiliated Workshops.

Z. q. Luo, W. k. Ma, A. M. c. So, Y. Ye, and S. Zhang. 2010. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Processing Magazine* 27, 3 (May 2010), 20–34. DOI: https://doi.org/10.1109/MSP.2010.936019

Tarek K. Refaat, Burak Kantarci, and Hussein T. Mouftah. 2016. Virtual machine migration and management for vehicular clouds. *Vehicular Communications* 4, Supplement C (2016), 47–56.

R. Roman, J. Lopez, and M. Mambo. 2018. Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges. *Future Generation Computer Systems* 78 (2018), 680–698. DOI: https://doi.org/10.1016/j.future.2016.11.009

R. Islam Rony, E. Lopez-Aguilera, and E. Garcia-Villegas. 2018. Access-aware backhaul optimization in 5G. In *Proceedings of the 16th ACM International Symposium on Mobility Management and Wireless Access (MobiWac'18)*. 124–127.

D. Samant and U. Bellur. 2016. Handling boot storms in virtualized data centers–A survey. *ACM Comput. Surv.* 49, 1, Article 16 (June 2016), 36 pages. DOI: https://doi.org/10.1145/2932709

Dimas Satria, Daihee Park, and Minho Jo. 2017. Recovery for overloaded mobile edge computing. *Future Generation Computer Systems* 70 (2017), 138–147. DOI: https://doi.org/10.1016/j.future.2016.06.024

Victor Soto, Robson E. De Grande, and Azzedine Boukerche. 2017. REPRO: Time-constrained data retrieval for edge offloading in vehicular clouds. In *Proceedings of the 14th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks (PE-WASUN'17)*. ACM, New York, 47–54. DOI: https://doi.org/10.1145/3134829.3134834

Peng Sun, Azzedine Boukerche, and Rodolfo W. L. Coutinho. 2019. A novel Cloudlet-Dwell-Time estimation method for assisting vehicular edge computing applications. In *Proceedings of the 2019 IEEE Global Communications Conference: Next-Generation Networking and Internet (Globecom2019 NGNI)*. Waikoloa.

Linlin Tang, Zuohua Li, Pingfei Ren, Jengshyang Pan, Zheming Lu, Jingyong Su, and Zhenyu Meng. 2017. Online and offline based load balance algorithm in cloud computing. *Knowledge-Based Systems* 138 (2017), 91–104. DOI: https://doi.org/10.1016/j.knosys.2017.09.040

A. N. Toosi, R. N. Calheiros, and R. Buyya. 2014. Interconnected cloud computing environments: Challenges, taxonomy, and survey. *ACM Comput. Surv.* 47, 1, Article 7 (May 2014), 47 pages. DOI: https://doi.org/10.1145/2593512

L. Vigneri, T. Spyropoulos, and C. Barakat. 2017. Quality of experience-aware mobile edge caching through a vehicular cloud. In *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'17)*. 91–98.

C. Wang, Y. Li, D. Jin, and S. Chen. 2016. On the serviceability of mobile vehicular cloudlets in a large-scale urban environment. *IEEE Transactions on Intelligent Transportation Systems* 17, 10 (2016), 2960–2970.

Zi Wang, Zhiwei Zhao, Geyong Min, Xinyuan Huang, Qiang Ni, and Rong Wang. 2018. User mobility aware task assignment for mobile edge computing. *Future Generation Computer Systems* (2018), DOI: https://doi.org/10.1016/j.future.2018.02.014

H. Wen, L. Yang, and Z. Wang. 2017. ParGen: A parallel method for partitioning data stream applications in mobile edge computing. *IEEE Access* PP, 99 (2017), 1–1. DOI: https://doi.org/10.1109/ACCESS.2017.2776358

Y. Wu and J. Zheng. 2020. Modeling and analysis of the uplink local delay in MEC-based VANETs. *IEEE Transactions on Vehicular Technology* (2020), 1–1. DOI: https://doi.org/10.1109/TVT.2020.2970551

P. Yang, Q. Li, Y. Yan, X. Y. Li, Y. Xiong, B. Wang, and X. Sun. 2016. Friend is treasure: Exploring and exploiting mobile social contacts for efficient task offloading. *IEEE Transactions on Vehicular Technology* 65, 7 (July 2016), 5485–5496. DOI: https://doi.org/10.1109/TVT.2015.2465392

H. Yao, D. Zeng, H. Huang, S. Guo, A. Barnawi, and I. Stojmenovic. 2015. Opportunistic offloading of deadline-constrained bulk cellular traffic in vehicular DTNs. *IEEE Trans. on Computers* 64, 12 (2015), 3515–3527.

X. Yu, R. W. L. Coutinho, A. Boukerche, and A. A. F. Loureiro. 2017. A distance-based interest forwarding protocol for vehicular information-centric networks. In *Proceedings of the 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. 1–5.

Q. Yuan, J. Li, Z. Liu, and F. Yang. 2016. Space and time constrained data offloading in vehicular networks. In *Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. 398–405.

Feifei Zhang, Jidong Ge, Zhongjin Li, Chuanyi Li, Chifong Wong, Li Kong, Bin Luo, and Victor Chang. 2018a. A load-aware resource allocation and task scheduling for the emerging cloudlet system. *Future Generation Computer Systems* (2018). DOI: https://doi.org/10.1016/j.future.2018.01.053

J. Zhang, H. Guo, J. Liu, and Y. Zhang. 2019. Task offloading in vehicular edge computing networks: A load-balancing solution. *IEEE Transactions on Vehicular Technology* (2019), 1–13. DOI: https://doi.org/10.1109/TVT.2019.2959410

K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang. 2017. Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading. *IEEE Vehicular Technology Magazine* 12, 2 (June 2017), 36–44.

K. Zhang, Y. Mao, S. Leng, A. Vinel, and Y. Zhang. 2016. Delay constrained offloading for mobile edge computing in cloud-enabled vehicular networks. In *Proceedings of the 2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM)*. 288–294. DOI: https://doi.org/10.1109/RNDM.2016.7608300

Wen-Li Zhang, Bing Guo, Yan Shen, De-Guang Li, and Jun-Ke Li. 2018b. An energy-efficient algorithm for multi-site application partitioning in MCC. *Sustainable Computing: Informatics and Systems* (2018). DOI: https://doi.org/10.1016/j.suscom.2018.02.008

P. Zhao, L. Feng, P. Yu, W. Li, and X. Qiu. 2017. A social-aware resource allocation for 5G device-to-device multicast communication. *IEEE Access* 5 (2017), 15717–15730. DOI: https://doi.org/10.1109/ACCESS.2017.2731805

Received November 2018; revised February 2020; accepted February 2020