Hindawi
Journal of Advanced Transportation
Volume 2019, Article ID 3548213, 21 pages
https://doi.org/10.1155/2019/3548213

Research Article

Sporadic Cloud-Based Mobile Augmentation on the Top of a Virtualization Layer: A Case Study of Collaborative Downloads in VANETs

Esteban Fernando Ordóñez-Morales,[1] **Martín López-Nores**,[2] **Yolanda Blanco-Fernández**,[2] **Efren Patricio Reinoso-Mendoza**,[1] **Jack Fernando Bravo-Torres**,[1] **José Víctor Saiáns-Vázquez**,[2] **José Juan Pazos-Arias**,[2] **Manuel Ramos-Cabrer**,[2] **and Alberto Gil-Solla**[2]

[1] *Universidad Politécnica Salesiana, Calle Vieja 12-30 y Elia Liut, Cuenca, Ecuador*
[2] *Atlantic Research Center for Information and Communication Technologies (AtlantTIC), University of Vigo, Spain*

Correspondence should be addressed to Esteban Fernando Ordóñez-Morales; eordonez@ups.edu.ec

Received 28 November 2018; Revised 28 January 2019; Accepted 27 February 2019; Published 21 March 2019

Academic Editor: Yair Wiseman

Current approaches to Cloud-based Mobile Augmentation (CMA) leverage (cloud-based) resources to meet the requirements of rich mobile applications, so that a terminal (the so-called *application node* or AppN) can borrow resources lent by a set of *collaborator nodes* (CNs). In the most sophisticated approaches proposed for vehicular scenarios, the collaborators are nearby vehicles that must remain together near the application node because the augmentation service is interrupted when they move apart. This leads to disruption in the execution of the applications and consequently impoverishes the mobile users' experience. This paper describes a CMA approach that is able to restore the augmentation service transparently when AppNs and CNs separate. The functioning is illustrated by a NaaS model where the AppNs access web contents that are collaboratively downloaded by a set of CNs, exploiting both roadside units and opportunistic networking. The performance of the resulting approach has been evaluated via simulations, achieving promising results in terms of number of downloads, average download times, and network overhead.

1. Introduction

The incorporation of on-board units (OBUs) into vehicles allows envisaging advanced information services grounded on vehicle-to-vehicle communications and the possibility of accessing the Internet via roadside infrastructure. These services allow (i) delivering information of interest between the occupants of the vehicles (e.g., chats among drivers, proactive organization of ride-sharing opportunities, selective distribution of personalized advertising, or collaborative downloading of web contents [1, 2]) and (ii) enabling intelligent transportation systems offering increased traffic safety, smart parking, or traffic flow optimization, just to name a few [3, 4]. Sharing and managing this kind of data requires the running of resource-intensive applications, and to this aim, both the vehicles' OBUs and the occupants' mobile devices need to be augmented to have extra processing/storage/downloading/sensing/... capabilities. In literature, most of approaches resort to *Mobile Cloud Computing* (MCC) to bring rich computational resources to mobile users, and more specifically to *Vehicular Cloud Computing* (VCC) when the augmentation resources are lent by other vehicles within a vehicular ad hoc network (VANET) [5]. Both paradigms are under the umbrella of the so-called *Cloud-based Mobile Augmentation* (CMA) where a set of nodes (hereafter called *collaborator nodes* or CNs) lend their own resources to augment the capabilities of the so-called *application node* (AppN). For that purpose, current CMA approaches leverage cloud-based resources to meet the requirements of rich mobile applications by offloading (part

of) their codes and the data required for their execution to a remote server. According to the taxonomy of [6], existing CMA solutions can be categorized into four main types as per the location and mobility of the augmentation clouds: *distant immobile clouds*, *proximate immobile clouds*, *proximate mobile clouds*, and *hybrid approaches* that combine the previous models [7, 8].

(i) The *distant immobile clouds* (MCC) are supported by public and private clouds that comprise a large number of stationary remote servers that are typically located far from vehicles. The distance to the base station, along with the interference caused by the presence of buildings or a high number of simultaneously connected users, leads to problems of high latency and low bandwidth that greatly degrade the experience of mobile users and hamper the deployment of non-delay tolerant information services [9, 10].

(ii) In order to tackle problems related to bandwidth and latency, some CMA solutions adopt *proximate immobile clouds* (typically named cloudlets) that involve stationary computers located in public places near the vehicles (e.g., parking lots, malls, and airports), whose resources are typically underutilized. The problem is that these approaches significantly limit the users' mobility because resource-intensive mobile applications can be just executed in the area around the cloudlet [11].

(iii) The most recent CMA approaches rely on *proximate mobile clouds* (VCC) where nearby vehicles lend their resources to other vehicles to perform intensive computations in a distributed manner [12]. Despite their name, the first approaches proposed in this regard disregarded the mobility of the augmentation nodes (by handling instead clusters of fixed handheld devices) due to the open challenges related to this trait (such as communication disruption as the users move and the consequent frequent interruptions of application execution [13]). More sophisticated approaches arose later in literature [14, 15]. However, in these solutions the management of the mobility continues being an open problem, due to the fact that they restrict the movements of the AppN and its collaborators (requiring, for instance, that all these nodes remain together during the augmentation process, which is interrupted when AppN and CNs move away) [16–18].

Bearing in mind the above limitations, in [19] we proposed an approach where the augmentation resources were shared/lent by a sporadic cloud of near vehicles that remain together during a certain period of time. For that reason, we coined our approach as *Sporadic Cloud-based Mobile Augmentation* (hereafter S-CMA), which is based on a new paradigm we have called *Sporadic Cloud Computing* (SCC). In other words, S-CMA is based on the SCC principles, in the same way that traditional CMA relies on MCC/VCC foundations. In particular, in [19] we envisaged the common foundations for the deployment of diverse "X"aaS service models over a VANET (such as *Networking as a Service* (NaaS), *STorage as a Service* (STaaS), *Computing as a Service* (CaaS), and *SEnsing as a Service* (SEaaS), just to name a few). Later, in [20] we focused on the particular refinements that, working on the top of the common basis, were required in a NaaS service. In this paper, we have extended that previous work developing a hybrid S-CMA approach that exploits both the augmentation capabilities of a sporadic cloud of on-move vehicles and the resources available at roadside-fixed infrastructure (through WiFi access points). Unlike existing CMA approaches, our sporadic cloud comprises an ad hoc cluster of mobile devices that do move freely, where (i) their (high) mobility is managed by a virtualization layer (named VaNetLayer) that engages the mobile nodes in collaboration to emulate a reliable infrastructure of stationary virtual nodes [21], and (ii) a seamless handover allows restoring the augmentation service transparently for the applications when AppN and CNs separate.

In our approach the web contents will be splitted into chunks that the CNs must download (from a remote server) and finally deliver to the AppN. To this aim, we exploit a NaaS model that supports both individualized access to web contents of interest for just one vehicle within the VANET (hereafter *individual content*) and access to *popular contents* that are appealing to all/most of the vehicles within the ad hoc network, thus leading to two download models which we will be referred to as *individual content download* (ICD) and *popular content download* (PCD), respectively. As a main difference, in PCD some collaborator nodes could have downloaded previously (for other interested AppNs) and locally stored some chunks of the requested popular contents, whereas in ICD (all the pieces of) the individual contents must be always retrieved from the Internet. In conclusion, we contribute with a new hybrid S-CMA approach for the deployment of a NaaS model for collaborative download of web contents in VANETs, working on the top of virtualization mechanisms that hide the complexity derived from the mobility of the augmentation cloud, which is an issue that, to the best of our knowledge, remains unresolved in the literature.

This paper is organized as follows. In Section 2 we present a review of existing approaches to collaborative download of web contents in VANETs, along with the mechanisms that are typically adopted to incentivize the involvement of nodes in augmentation tasks. Next, Section 3 summarizes the main foundations of the virtualization layer (VaNetLayer) and the routing protocol (VNIBR) on which our hybrid S-CMA approach works. The S-CMA mechanisms envisaged for the deployment of our NaaS model will be presented in Section 4. The experiments carried out to assess the performance of our proposal in terms of number of downloads, download time, networking overhead, and some augmentation-specific metrics will be described in Section 5. Lastly, Section 6 concludes the paper and points out some lines of further work.

2. Related Works

In literature, we can found many approaches to collaborative download of popular content from the web that are based on V2I communications. As we will detail in Section 2.1, some of them take advantage of the formation of groups vehicles that travel together on the road in order to download content and then share it [22–25], whereas other works are supported by vehicles parked on the streets [12, 26–29] or even vehicles that form clusters according to their geographical positions [16, 30]. Beyond the technical challenges, these approaches require incentive mechanisms to promote the collaboration of the nodes during the download processes (Section 2.2).

2.1. Collaborative Download. SPAWN was one of the first protocols for collaborative download of Internet contents in VANETs [31]. SPAWN manages groups (swarms) of mobile nodes that get different chunks of web contents (seeds) during periods of connectedness. A gossip mechanism propagates content availability information within the swarms, so that the nodes can get chunks they are missing from one another. Chunks are transmitted by TCP at the transport layer, whereas UDP is adopted for gossip messages. CarTorrent [23] was proposed as an evolution of SPAWN, with a different strategy to select chunks to download, a refined gossip mechanism, and additional controls to use only UDP at the transport layer (due to the bad behavior of TCP in case of frequent packet losses). Later, CodeTorrent proposes new mechanisms to speed up the content dissemination and be more resilient against fast mobility, relying on single-hop unicast with overhearing [24]. With a slightly different perspective, MobTorrent uses Wireless Wide-Area Networks 2G/3G/4G (WWANs) to provide a communication channel, so that (i) the vehicles can upload information to the access points in order for the remaining vehicles to be able to download it when they are within their coverage area, and (ii) the vehicles can offer themselves as collaborators to carry the content they have previously downloaded for other vehicles [25].

In [32] Huang et al. proposed a k-hop bandwidth aggregation scheme for cooperative video streaming in a vehicular environment, by using 3G/3.5G and Dedicate Short-Range Communications (DSRC) ad hoc networks, along with nearby vehicles as collaborators nodes. In [16] Firooz and Roy present a more restrictive cooperative download scheme, which requires that the vehicles remain close from each other and keep the same route on the way. The same constraints are valid for the approach described in [33], where Wang et al. describe a mechanism for downloading popular content by deploying a P2P network, where the content to be downloaded is divided into chunks that are distributed over the VANET and finally delivered to all the vehicles.

In [26] Liu et al. presented ParkCast, an approach that exploits the networking capabilities of parked vehicles at roadside. These nodes are grouped in a cluster, with one of the vehicles being selected as coordinator for data transmission. The vehicles can both upload information to the cluster of parked nodes and download data from it, as long as they stay within its coverage area. Similarly, in [30] Huang and Wang presented a cell-based clustering scheme, named Efficient Collaborative Downloading Scheme (ECDS), that works on distributing popular content in urban vehicular networks. In ECDS the vehicles join in different cells according to their geographical positions. Treating the cells as nodes, their positions are fixed, which greatly simplifies the modeling of the VANET. When the vehicles are outside the coverage of the roadside units, they form a P2P network and collaboratively exchange chunks to complete the popular content dissemination.

2.2. Incentive Schemes for Collaboration. Usually, the above approaches work in tandem with different mechanisms for promoting the collaboration among the vehicles, with the goal of improving the performance in face of topological changes in the ad hoc network. Even though some existing collaboration mechanisms are categorized as per the kind of ad hoc network where they are deployed (MANETs/VANETs) and the type of transmitted information (non-delay/delay tolerant), there exist two main trends: (i) *monetary reward systems* and (ii) *reputation-based systems*. The first ones handle virtual money that can be finally exchanged for real money, and resort to a central entity that manages the (economic) transactions made by the collaborator nodes. In the reputation-based collaboration system, the collaborators gain reputation scores as they collaborate in the network.

As examples of monetary reward systems we can cite the approach by Chen et al. presented in [34], where the authors propose a collaborative mechanism based on coalitional game theory that rewards the cooperation among the nodes involved in a routing protocol. This approach includes a virtual credit center [35] that keeps updated the amount of virtual money of each node, whose value is increased as the nodes collaborate in the forwarding of messages. In the same line, in [36] Ananthanarayanan et al. propose an approach to collaborative download called COMBINE, which adopts a cooperation solution with monetary incentives for nodes that sell their (unused) bandwidth to augment the capabilities of other nodes within the network, so that the collaborators offering the best prices can be chosen. In COMBINE the transactions and payments among nodes are protected by security mechanisms based on public and private keys. In [37], Caballero et al. propose a mechanism where the reward for each collaborator node is computed once the content they have downloaded is finally delivered to the application node, by considering their percentage of participation in this process.

Regarding reputation-based collaboration systems, Dotzer et al. proposed in [38] a distributed approach (named VARS) where a number of vehicles move at high speeds. VARS works with several areas: (i) an *event area* that is the physical region where an event occurs, (ii) the *decision area* that defines the integrity of the involved messages, and (iii) the *range*, which specifies how far these messages can be distributed. The reputation score of each node is computed by checking the integrity of the messages sent by the collaborators and considering also parameters related to trust and opinions given by the rest of nodes involved in the forwarding process. As another sample,

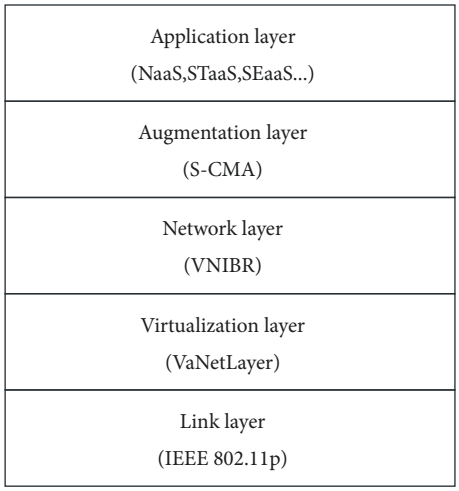

Figure 1: Protocol stack of our approach.

note the approach proposed by Li in [39] that handles a comprehensive announcement mechanism that enables assessing the reliability of the messages, by deciding to forward (or not) a message as per the reputation of the sender node. These scores are computed staring from the number of reliable messages transmitted in the past and are kept in a centralized server that is connected to each node via WiFi access points located in strategic locations (e.g., gas stations).

3. Background on VaNetLayer and VNIBR

Most of the approaches mentioned in the previous section have problems when it comes to handling the mobility of the vehicles due to the constant changes in the VANET topology. This causes disruption in the communications, packet losses, low throughput, overhead of the routing protocols, etc. In our case the problems stemmed from the mobility are managed by a *virtualization layer* named VaNetLayer, which is a cluster-based approach where the mobile nodes collaboratively create an infrastructure of *static virtual nodes* (VNs) to ease the routing problem and the maintenance of *Persistent State Information* (PSI) in the area covered by VANET, notwithstanding the mobility of the real physical nodes (PNs). The protocol stack of our approach is depicted in Figure 1, where the link layer supports the protocol IEEE 802.11p that has been specifically developed for vehicular networks. S-CMA must provide transport-layer mechanisms to coordinate the devices that are willing to share their capabilities for augmenting other terminals of the VANET, by dealing with virtual nodes of the VaNetLayer (Section 3.1). This way, S-CMA enables the execution of resource-intensive applications to improve the experience of the users, without caring about the mobility of the nodes included in the augmentation sporadic cloud. In fact, the mobility-awareness is provisioned from the virtualization layer, which ensures a reliable data exchange thanks to the collaboration with the routing protocol VNIBR (Section 3.2).

3.1. VaNetLayer. As depicted in Figure 2, the VaNetLayer divides the geographical area of the VANET into regions following an intersection-based layout, placing one VN at each crossroad and covering the road segments in between with equispaced VNs. We denote these regions as *crossroad regions* and *road segment regions*, respectively. In each region, one or several PNs are chosen as *leaders* to take charge of packet reception, buffering, and forwarding in the communication with other VNs. In turn, a subset of nonleaders work as *backups* to maintain replicas of PSI, so that the VNs can be fault-tolerant even when individual PNs fail or leave the region, as long as there remains at least one PN.

Apart from backups that maintain PSI replicas, VaNetLayer has *snapshot mechanisms* to preserve the PSI when the regions are left without physical nodes (fallen VNs). The *snapshot mechanisms* work supporting the information of the VNs located in the intersections, through the neighboring VNs of the intersection. This mechanisms allow upper layers to continue to work, even though the VNs of the intersections are down.

The VNs are addressed by class E IP addresses ("Experimental", between 240.0.0.0 and 255.255.255.255) and their size is determined by the communications range of the underlying link layer protocol, to guarantee that a PN in any position in one region can communicate directly with any other PN in a neighboring region.

In [40], we have tested the refinements envisaged in our VaNetLayer through an application for accessing web contents, where the nodes download contents directly through WiFi access points (APs). In our experiments, we adopted the routing protocol *Virtual Node Ad hoc On-demand Distance Vector* (VNAODV) [21], which is a virtualization-driven version of the classical protocol AODV [41] that routes traffic through virtual nodes (instead of physical nodes like in AODV). The tandem VaNetLayer-VNAODV was compared with existing approaches to collaborative download of web contents, such as CarTorrent [23] and CodeTorrent [24]. Briefly speaking, the simulations results showed the following benefits of the VaNetLayer (see details in [40]):

(i) Virtualization overhead is similar to the amount of control information handled in CarTorrent.

(ii) The packet delivery ratios are high (comparable to CodeTorrent).

(iii) The approach scales well with the number of nodes within the VaNetLayer.

There results confirm that the efforts establishing and maintaining the virtual node infrastructure enable reliable communications, where the VNs can be seen as stationary sources of computation and resources, thus leading to greater levels of reliability for the upper layers (routing and sporadic Cloud-based Mobile Augmentation).

3.2. VNIBR. The interface between the VaNetLayer and the network layer exposes the notion of regions, the role (leader or backup) played by a PN at each moment, and functions to send/receive messages and to get/set/check the PSI. This way, it is possible to adapt existing routing algorithms to lean on

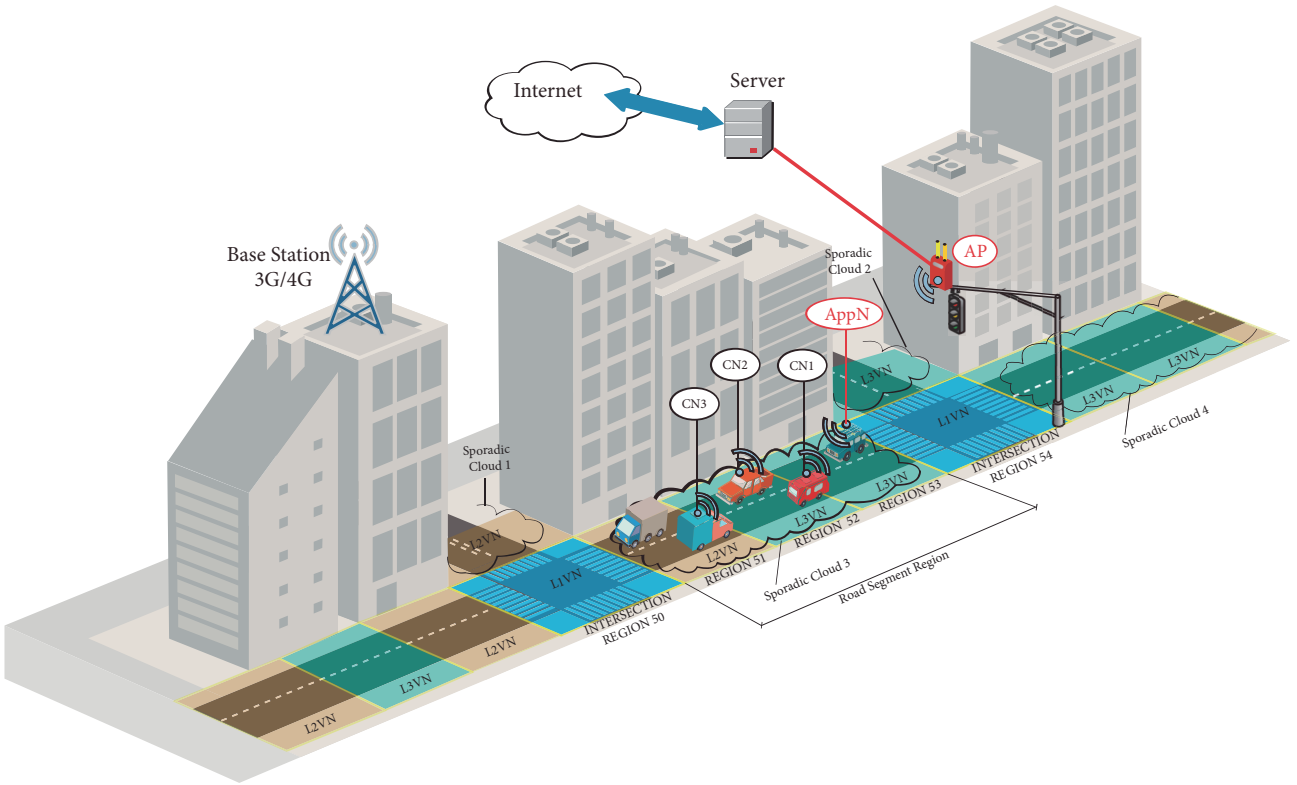

Figure 2: Distribution of static virtual nodes and definition of L1VN, L2VN, and L3VN in our VaNetLayer.

the VNs as stable routing entities or even as destinations for geocasting. In particular, we have developed a protocol that enables an efficient combination of topological and geographical routing [42], which is called VNIBR (Intersection-Based Routing on Virtual Nodes). In this protocol, we identify three types of routing entities (named L1VNs, L2VNs, and L3VNs) that are depicted in Figure 2 (in colors blue, brown, and green, respectively):

(i) Level 1 entities (L1VNs) are the VNs placed at the intersections. This is where the routing decisions are made, using the procedure we will describe later in this section. Routing tables are transparently kept by the VaNetLayer as PSI, with entries indicating which is the next road segment (identified by the L1VN at the other end) that the packets must traverse in order to reach the corresponding destination. Note that the L1VNs that are located at a crossroad where there is a fixed AP are permanently available virtual nodes, with no downtimes thanks to the WiFi connectivity.

(ii) Level 2 entities (L2VNs) are the VNs neighboring an intersection where an AP is not available. These VNs start forwarding packets along a road segment as mandated by the neighboring L1VN, irrespective of whichever PN actually does the transmission. Besides, L2VNs act also as backing entities, trying to relay packets onto other road segments during downtimes of the neighboring L1VNs.

(iii) Finally, as depicted in Figure 2, level 3 entities (L3VNs) are the VNs that are either (i) in intermediate position of road segments or (ii) in a region neighboring an intersection with an AP. These nodes simply relay packets from one side to the other, again irrespective of specific PNs.

The L2VNs and L3VNs forward the packets from one end of the road segments to the other, without any other processing than setting a delivery bit in the header to 1 if the destination PN is within the current region. Due to the reactive nature, communication routes in VNIBR are created only when a source PN needs to send a packet but it does not know a route to the intended destination PN. VNIBR handles the following messages:

(i) *M_Hello*: This message is steadily exchanged between L1VNs which are one road segment away from one another to keep track of (i) the connectivity conditions they provide and (ii) the average number of physical nodes in its virtual region, calculated using an *exponentially weighted moving average* (EWMA). This way, when an L1VN receives a *M_Hello* message, it assigns a QoS value to that link, pondering the L1VN-to-L1VN transmission delay (again, filtered by EWMA), the average number of PNs in the intermediate VNs, and the amount of data traffic going through that road segment (recall Figure 2). This information is kept and updated in the two L1VNs

that delimit the road segment by the periodic transmission of this type of messages. If a road segment does not provide connectivity to the other end, as determined by the exchange of *M_Hello* messages, it is skipped. The process of exchange of *M_Hello* messages starts when a timer called *T_Hello* ends. This timer is initialized when the L1VN gets up for first time or it is restored after a fall.

(ii) *M_RouteRequest*: This message is used by a PN that needs to send a data packet to the other node within VANET. This PN sends this message to the L1VNs that delimit its road segment. Each one of those L1VNs puts its ID as the first element of a L1VN list in the *M_RouteRequest* message header and sends a copy to other L1VNs which are one road segment away. A broadcast ID is included in the periodically transmitted *M_RouteRequest* message to prevent any L1VN from transmitting the same packet more than once. Anyway, the flooding implies very little overhead because it involves only the leaders of the traversed VNs (unless some backup PNs have to assist as a result of packet losses). A route is established (i) when the *M_RouteRequest* message reaches an L1VN that knows a valid route to the destination PN, (ii) when an L1VN receives the *M_RouteRequest* message with the delivery bit set to 1, meaning that the destination PN is in the road segment just traversed, or (iii) when an L1VN receives the *M_RouteRequest* message with the delivery bit set to 0 but the destination PN is in the current intersection.

(iii) *M_RouteReply*: This message is used when a created route travels along the backpath, allowing the L1VNs on the way to set up the path to the destination PN in their routing tables.

Once a route between source and destination PNs has been established, the data packets are sent VN by VN via unicast transmissions between leader nodes. When the VaNetLayer detects a failure (e.g., when it is not possible to resolve the MAC address of the next hop node, when the RTS/CTS mechanism of IEEE 802.11 cannot reserve the shared channel, or when no acknowledgment for a data packet can be received and retransmission attempts also failed), different mechanisms to report the error or even to endeavor a local repair can be exploited. Further details (out of the scope of this paper) can be found in [43].

The performance of VNIBR protocol has been evaluated by the simulation presented in [44]. In particular, in this work we compared our VNIBR protocol against AODV and VNAODV. Briefly speaking, the simulation results showed the following:

(i) Both versions of VNIBR provide better performance than AODV and VNAODV in terms of overhead.

(ii) Regarding packet delivery ratios, reactive VNIBR outperforms proactive VNIBR in scenarios with sparse vehicular traffic density, where the low mobility of vehicles leads to more stable and reliable communications between L1VNs.

(iii) Lastly, the results for end-to-end delays suffered by the packets that do reach the intended destinations show that the both the proactive and reactive versions of VNIBR are faster than traditional routing protocols.

4. S-CMA: Sporadic Cloud-Based Mobile Augmentation

By virtue of its hybrid nature, the S-CMA approach that supports our NaaS model exploits both the bandwidth lent by an ad hoc cluster of close on-move vehicles (*proximate mobile sporadic cloud*) and the availability of roadside units. In our proposal, several AppNs can be augmented (to access individual/popular web contents) by a set of CNs that are in charge of downloading (from a remote server) and finally delivering all the chunks of the requested content. Specifically, the server computes the number of chunks per content, thus acting as an additional augmentation source. The chunks are processed as binary data and a hash code is used for integrity checking at the receiver side. Besides, the server may choose to apply compression techniques (e.g., based on Lempel-Ziv or Burrows-Wheeler algorithms) to reduce the size of the chunks to be downloaded—as in [45, 46], the selection of one technique or another can consider parameters such as speed and compression efficiency, quality of the communication links (latency, available bandwidth,...), and type of data. Anyhow, the availability of the downloaded content is confirmed from the server via PUSH notifications [47] which report on the URL, the number of chunks, and their respective identifiers. (If the server cannot find the requested content, a PUSH notification "*file not found*" is sent to the application layer of the AppN.)

Our S-CMA mechanisms need to collect information about the amount of resources that each CN is willing to share to augment the AppN. To this aim, we have envisaged two main processes to meet the requirements stemmed from the highly changing topology of VANETs, which are periodically executed to maintain fresh information:

(i) *Resource report:* This process allows the CNs to send information about the augmentation resources lent. These resources include both the available bandwidth and the possible chunks of popular contents that each CN has previously downloaded (after being requested by other AppNs) and locally stored to be shared with other interested vehicles in the ad hoc network. This way of advertising the availability of popular chunks in some CNs prevents redownloading of these contents, which brings significant benefits in our NaaS model, as evidenced in the simulations presented in Section 5.

The information about the augmentation resources is reported by each CN through a *M_ResourceInfoFromCN* message, which is sent to the leader of its current region when (i) the CN enters a new region (except at the intersections), and (ii) the CN reports

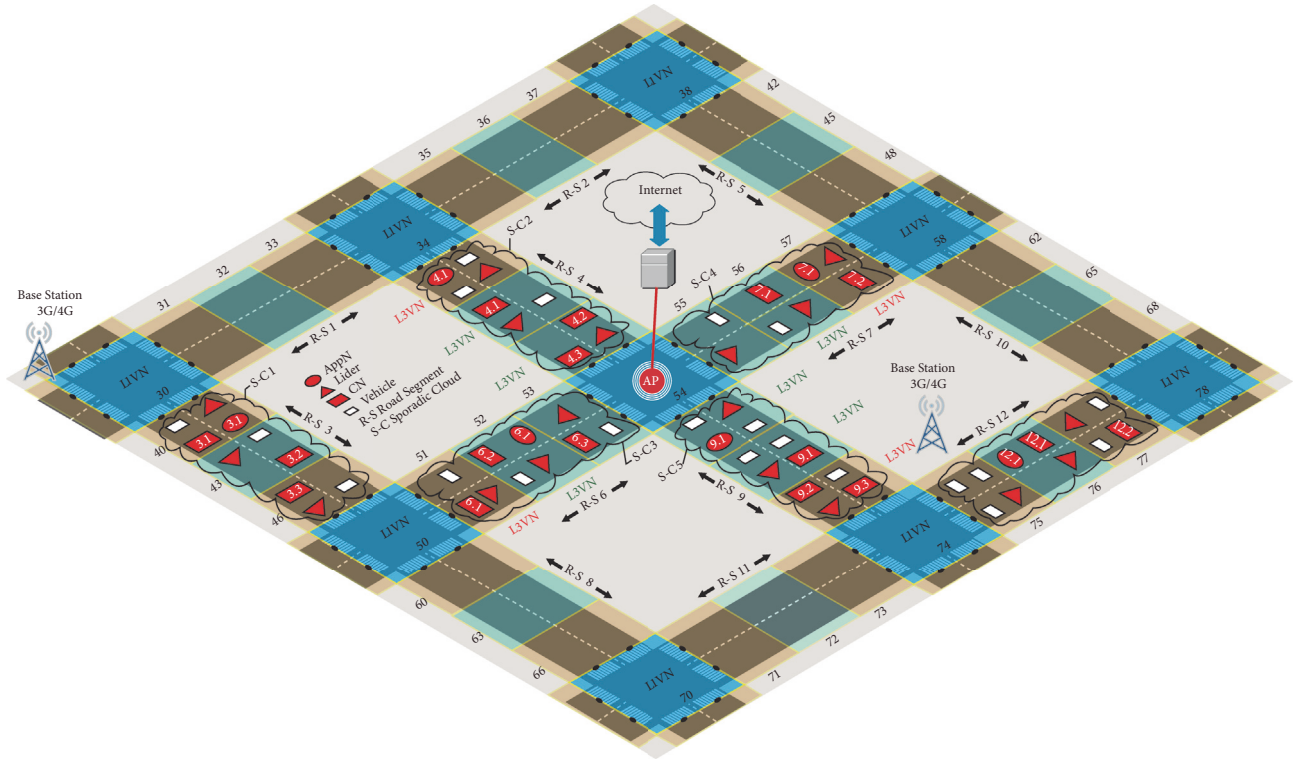

Figure 3: Possible connectivity options of AppNs in the NaaS model supported by our S-CMA mechanisms.

on the availability of new augmentation resources after finishing a previous download for an AppN.

(ii) *Resource discovery*: This process consists of exchanging *M_ResourceDiscovery* messages between the L1VNs that are one road segment away from one another, in order to collect the PSI about the available bandwidth in each VN deployed over the road segment. The process *resource discovery* starts once the *T_Discovery* timer has expired, which is initialized when the L1VN starts to work for first time or is restored after a fall.

Since the augmentation process is triggered when the AppN requests a particular (individual/popular) content, this node needs to get information about the amount (and IDs) of the chunks to be downloaded by its CNs. As this information is available at the remote server, we can identify two possible *connectivity scenarios* as per the location of the AppN:

(i) *The AppN is located in a road segment that is delimited by an intersection where an AP is located.* Since the AP covers the four road segment regions neighboring the intersections, in this scenario it is possible that the AppN cannot connect to the remote server. For instance, the AppN6.1, AppN7.1, and AppN4.1 (located in road segments 6, 7, and 4 in Figure 3, respectively) need to form a sporadic cloud (by the procedure we will describe later in this section) to find CNs that retrieve information about the number of chunks to be downloaded. To this aim, the AppN communicates with the L1VN at the intersection where the AP is located, which maintains information about the availability of possible CNs and their respective augmentation resources. After receiving the requests from the CNs, the server allocates among them the corresponding download tasks. However, it is also possible that the AppN is located in a road segment region neighboring the intersection, connected via the AP (e.g., AppN9.1 in Figure 3). In this case, the AppN can retrieve from the server information about the amount of chunks to download. Next, the AppN triggers an augmentation request in order to form the sporadic cloud and ask each CN to download a given number of chunks (computed by the procedure we will describe in Section 4.5).

(ii) *The AppN is in a road segment whose (two) delimiting intersections do not have an AP.* As shown in Figure 3, in absence of a nearby AP, the AppN3.1 and AppN12.1 require that an augmentation request is triggered (to the closest L1VNs) to find CNs that get information from the server, which allocates again the download tasks among the collaborators without the participation of the AppN.

The augmentation procedure described next in this section will be common to the above two connectivity scenarios. In particular, the deployment of the augmentation sporadic mobile cloud is managed by different types of events, messages exchanged between AppN and CNs (see Table 1), timers to control the download tasks performed by each

Table 1: List of messages used in our S-CMA approach.

Used by	Messages	Description
CN	*M_ResourceInfoFromCN*	Used to send the resource information from a CN to a VN.
AppN	*M_SearchRequest*	Allows requesting a CN to perform content search, when an AppN is not connected to the Internet.
AppN	*M_ResourceRequest*	Used to request augmentation resources (bandwidth in NaaS).
AppN	*M_CloudResourceInfo*	Contains information about the available bandwidth for the augmentation process.
AppN	*M_WithoutCollaborators*	Indicates that there are no available augmentation resources.
AppN	*M_InsufficientResources*	Indicates that there are no enough augmentation resources.
AppN	*M_UnavailableAugmentationService*	Indicates to the application layer that the augmentation service is not available.
AppN	*M_CloudRelease*	Indicates the release of a sporadic cloud.
AppN	*M_SendTaskToCN*	Allows to send download tasks to CNs.
CN	*M_AcceptedTaskFromCN*	Allows the CNs to confirm that they accept to download contents for the AppN.
CN	*M_CompleteTask*	Allows the CNs to send chunks of the AppN-requested content.
AppN	*M_CompleteTaskACK*	Acknowledgement that the chunks have arrived correctly from the CN to the AppN.
CN	*M_IncompleteTask*	Used to report that a download task cannot continue.
AppN	*M_IncompleteTaskACK*	Acknowledgement that a download task cannot continue in a CN.
AppN	*M_TaskInfo*	Allows to send the information of completed/incomplete download tasks to the application layer of the AppN.

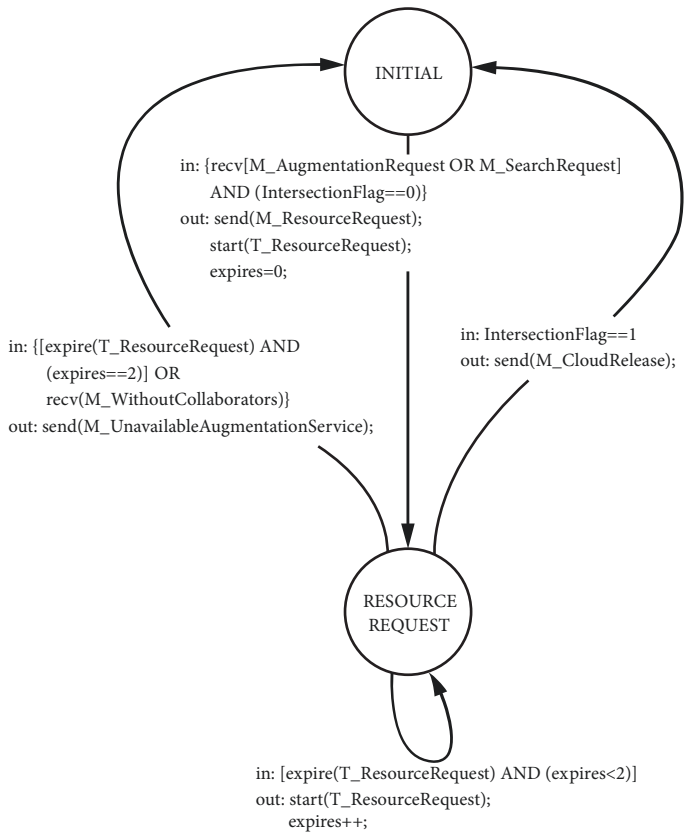

Figure 4: Transition from INITIAL to RESOURCE REQUEST state.

CN (Table 2), flags that provide information to the nodes (Table 3), and a set of possible states for each node.

We start by describing the possible states and the conditions that cause their respective transitions:

(i) INITIAL: This is the state that all the nodes start from.

(ii) RESOURCE REQUEST: This state indicates that the AppN requests augmentation resources in order to increase its downloading capabilities.

(iii) TASK DISTRIBUTION: The goal here is to allocate the download tasks among the CNs, by deciding how many chunks of the web content will be downloaded by each one of them.

(iv) COLLABORATION: Once the augmentation tasks have been accepted by the CNs, these nodes start to collaboratively download chunks from the server.

(v) TASK RECEPTION: In this state, the AppN waits for receiving the downloaded chunks in order to deliver them to the application layer.

4.1. INITIAL ⟶ RESOURCE REQUEST. When an AppN is located in a road segment region (which is denoted in Figure 4 as *IntersectionFlag=0*), it must get information about the amount of possible CNs available throughout that road segment. To this aim, the application layer of the AppN sends a *M_AugmentationRequest* message to the S-CMA layer (recall Figure 1), which causes our augmentation mechanisms to send a *M_ResourceRequest* message to the L1VN of the closest intersection. Next, the AppN changes from INITIAL to RESOURCE REQUEST state and waits for responses from the L1VN during the time set in the *T_ResourceRequest* timer, as shown in Figure 4.

After the reception of this request, the node L1VN checks the possibility of augmenting the AppN by exploiting the bandwidth borrowed from the CNs located in the same region segment as the AppN:

(i) If there are no CNs, L1VN reports the AppN via a *M_WithoutCollaborators* message. Next, the AppN switches again to INITIAL state and its application layer is notified by a *M_UnavailableAugmentationService* message.

(ii) Otherwise, L1VN reports on the amount of available CNs by a *M_CloudResourceInfo* message. If it does not arrive before the *T_ResourceRequest* expires three times, the AppN changes from RESOURCE REQUEST to INITIAL state. Otherwise, the AppN changes to TASK DISTRIBUTION state, starts a *T_TaskDistribution* timer, and begins to allocate among the CNs the chunks to download.

4.2. RESOURCE REQUEST ⟶ TASK DISTRIBUTION ⟶ TASK RECEPTION. In this point of our augmentation process, we can identify four different scenarios:

Table 2: List of timers used in our S-CMA approach.

Used by	Timers	Tune time to:
AppN	*T_ResourceRequest*	Wait for information about available augmentation resources.
AppN	*T_TaskDistribution*	Allow allocating download tasks among the CNs.
AppN	*T_IntersectionReception*	Wait for the tasks performed by the CNs when an AppN enters an intersection.
AppN	*T_Reception*	Wait for the chunks of content downloaded by the CNs.
CN	*T_CompleteTaskACK*	Wait for ACK of one or more completed chunks that have been sent from the CNs to the AppN.
CN	*T_IncompleteTaskACK*	Wait for ACK of a download task that cannot continue.
AppN	*T_GuardTime*	Wait a *Guard Time* for chunks sent by the CNs after receiving *M_CloudRelease* from the AppN.

Table 3: List of flags used in our S-CMA approach.

Used by	Flags	Flag=1 indicates that:
AppN/CN	*IntersectionFlag*	A node is at an intersection.
AppN	*DistributionFlag*	The distribution process of the download tasks were successfully completed.
CN	*FinishFlag*	The download task performed by a CN ended successfully.
CN	*AvailableResourcesFlag*	A CN has available augmentation resources.
CN	*AugmentationAgreeFlag*	A CN is willing to collaborate by lending its resources.

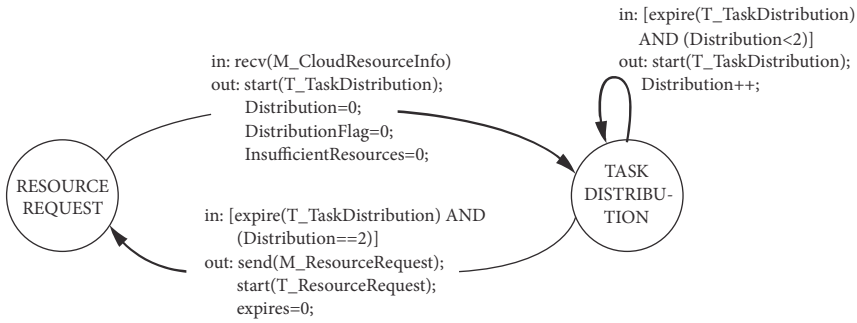

Figure 5: Transition from RESOURCE REQUEST to TASK DISTRIBUTION.

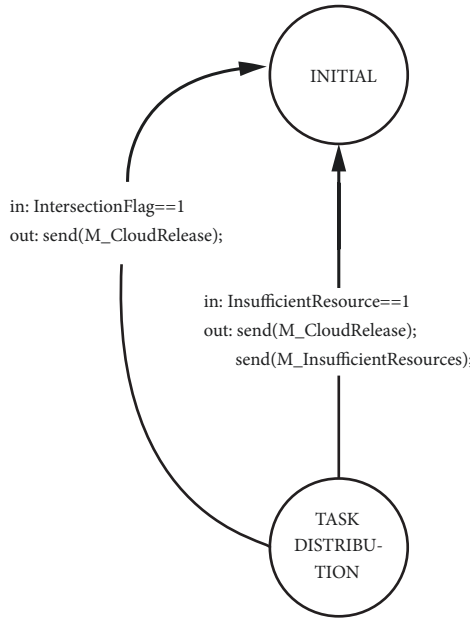

Figure 6: Transition from RESOURCE REQUEST to DISTRIBUTION state.

(i) First, if the *T_TaskDistribution* timer expires three times before concluding the chunk allocation, the AppN returns to RESOURCE REQUEST state and sends again the *M_ResourceRequest* message to the closest L1VN, as depicted in Figure 5. This action is required due to the fact that the availability of augmentation resources changes as CNs move (e.g., when the collaborators become inactive or leave a sporadic cloud after entering a crossroad).

(ii) The second possibility happens when the AppN finds that the CNs' bandwidth (reported by L1VN) is not enough for downloading the content. Then, as shown in Figure 6, a *M_CloudRelease* message is broadcast (in order to remove the resource request in the L1VN) and the application layer is also notified via a *M_InsufficientResources* message. Otherwise, the process continues in the AppN.

(iii) In this point, it is also possible that the AppN enters an intersection region (recall Figure 2) before concluding the chunk allocation among its CNs. In this case, the augmentation request must be deleted from the sporadic cloud that the AppN has just left, and

triggered again in the cloud corresponding to the new road segment that this node will traverse after leaving the intersection. To this aim, the AppN broadcasts a *M_CloudRelease* message before returning to the INITIAL state, as depicted in Figure 6. After receiving this message, the CNs can release their current augmentation resources which can be used to serve new augmentation requests from other AppNs, in a completely transparent way for the application layer.

(iv) The last possibility is that the chunk allocation finishes in time while the AppN is located in a road segment (which is denoted in Figure 7 as *DistributionFlag=1*). In this situation, the AppN sends the download tasks to the CNs of the sporadic cloud via *M_SendTaskToCN* messages, including the URL, the number of chunks, and their respective identifiers. Next, the AppN changes to TASK RECEPTION state and waits for CNs to confirm if they agree to download the corresponding pieces of the content.

4.3. TASK RECEPTION ⟶ INITIAL. The download tasks can be completed, not completed, or even rejected by the CNs of the sporadic cloud. We consider that a CN has completed its task when all the requested chunks have been successfully downloaded and delivered to the AppN. As depicted in Figure 8, as the AppN receives the notifications of the pending tasks (including completed, incomplete, and not accepted) from the CNs, it sends the corresponding information to the application layer (via a *M_TaskInfo* message for each task). The application layer receives the chunks, decompresses them if necessary, and, according to their chunk identifier, restores the original information:

(a) On the one hand, if the application layer requires more CNs to finish the incomplete and rejected downloads, a new augmentation process can be triggered following the guidelines described along this section. Note that this seamless handover is completely transparent for the application layer, so that the mobile applications of the users can continue working without noticing the transition between different augmentation clouds and without suffering any disruption (as long as there exist CNs willing to lend augmentation resources).

(b) On the other hand, if the CNs have completed all their download tasks, the AppN changes from TASK RECEPTION to INITIAL state. However, if some chunks still have not been delivered by the CNs before the (i) *T_Reception* timer expires two times or (ii) the AppN enters in an intersection, the AppN broadcasts a *M_CloudRelease* message over the sporadic cloud in order that the CNs cancel immediately their in-progress download tasks. In both cases, a *T_GuardTime* is set to wait for the missing chunks before sending the *M_CloudRelease* message. As shown in Figure 8, when this timer expires, the AppN returns to INITIAL state and the augmentation process ends.

4.4. INITIAL ⟶ COLLABORATION. The last state of our approach is associated with the CNs involved in the collaborative downloading process. In particular, as presented in Figure 9, a CN changes from INITIAL to COLLABORATION state after the reception of the *M_SendTaskToCN* message from the AppN through leader of its current region. Then, the CN confirms the acceptance of the download task by sending to the AppN a *M_ AcceptedTaskFromCN* message. As the CNs get the requested chunks, they send them to the AppN through a *M_CompleteTask* message.

According to what we explained in previous sections, the CNs must submit the intermediate results of the download tasks that had not been completed yet when they (i) hear the *M_CloudRelease* message, (ii) enter an intersection, or (iii) decide to stop being collaborators. To this aim, the CNs send the *M_IncompleteTask* message to the AppN. Since this message carries very relevant information (identifying missing pieces of the contents), the CN waits until the AppN acknowledges the reception. We have defined the *M_CompleteTaskACK* and *M_IncompleteTaskACK* messages, and the *T_CompleteTaskACK* and *T_IncompleteTaskACK* timers for this purpose.

Having delivered the pending tasks, the CNs release the involved resources and report to their VN on the new availability of their augmentation capabilities (via a *M_ResourceInfoFromCN* message). This information is also updated in the L1VNs located in the intersections thanks to the periodic sending of the *M_ResourceDiscovery* messages (recall the processes of *resource report* and *resource discovery* described in Section 4). Lastly, as shown in Figure 9, the CN returns to INITIAL state and waits to lend again its bandwidth when requested by an AppN.

The overall state machine that supports our augmentation mechanisms is shown in Figure 10, compiling all the transition among states described throughout this section.

4.5. How to Allocate Chunks to Collaborator Nodes. According to the two connectivity scenarios detailed at the beginning of Section 4, allocating the chunks that each CN must download is a task that can be performed by either the AppN or the remote server itself. In our augmentation mechanisms, this procedure considers two parameters that are reported from the CNs:

(i) The first parameter is the average time a CN will stay within the same sporadic cloud as the AppN, which is relevant because that collaborator cannot continue lending its bandwidth after leaving the road segment (triggering then a seamless handover for the applications). This parameter will be denoted as $RSAT_{CN_i}$ (Road Segment Average Time) for the i-th collaborator.

(ii) The second parameter is the bandwidth lent by each CN to the collaborative download process (denoted as BW_{CN_i}).

Obviously, the $RSAT_{CN_i}$ value depends on the average speed of CN_i within the corresponding segment road (denoted as $AvSpeed_{CN_i}$ in (1)) and on its location with

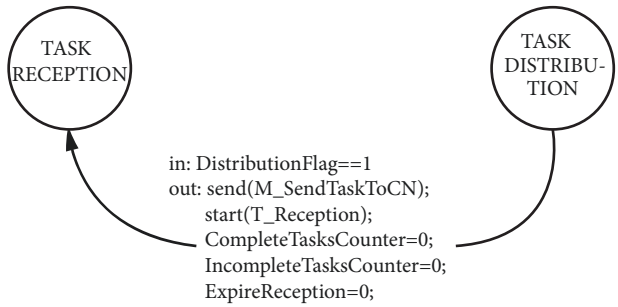

Figure 7: Transition from TASK DISTRIBUTION to TASK RECEPTION state.

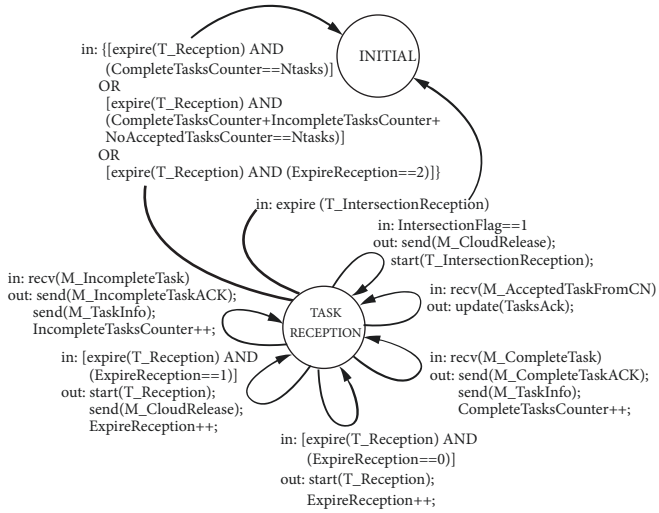

Figure 8: Transition from TASK RECEPTION to INITIAL state.

respect to the intersection ($Position_{CN_i}$). The position is computed resorting to a Haversine formula to compute the distance between CN_i's geographical location (i.e., GPS coordinates) and the reference points delimitating the next intersection region (see Figure 2).

$$RSAT_{CN_i} = \frac{Position_{CN_i}}{AVSpeed_{CN_i}} \quad (1)$$

Each CN periodically obtains new samples of the values needed by (1) and sends them to the corresponding VN through a *M_ResourceInfoFromCN* message. According to what we mentioned in the process *resource discovery*, this information is finally included in the *M_ResourceDiscovery* message used for (i) gathering information about the bandwidth (and the amount of chunks of popular contents) available in the CNs that are located along the segment road and (ii) keeping this information updated in the L1VNs of each intersection. Starting from the *RSAT* and BW_{CN_i} values of each collaborator and the chunk size (defined by the server through the compression process established), the AppN computes via (2) the number of chunks that CN_i will download (denoted as $\#chunks_{CN_i}$).

$$\#chunks_{CN_i} = \frac{BW_{CN_i} \cdot RSAT_{CN_i}}{chunk\_size} \quad (2)$$

As per (2), the longest download tasks are carried out by the CNs (i) with a high bandwidth and (ii) that are located at the beginning of a road segment, since these collaborators (according to their direction) take longer to reach an intersection. On the contrary, the CNs that are near the intersections take charge of shorter downloads because they remain for less time within the augmentation sporadic cloud formed in that road segment (recall Figure 3).

The outputs from (1) and (2) are used only locally and not aggregated in any way. Since the periodical samples may be taken at any point of a road segment or intersection, with vehicle speeds conditioned by a number of circumstances, the aggregates would be essentially random.

4.6. How to Incentivize the Collaboration among Nodes. In our proposal, the collaboration among the nodes involved in

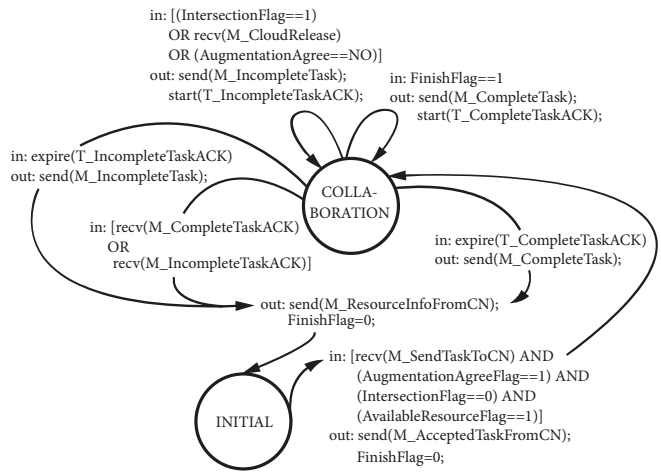

Figure 9: Transition from INITIAL to COLLABORATION state.

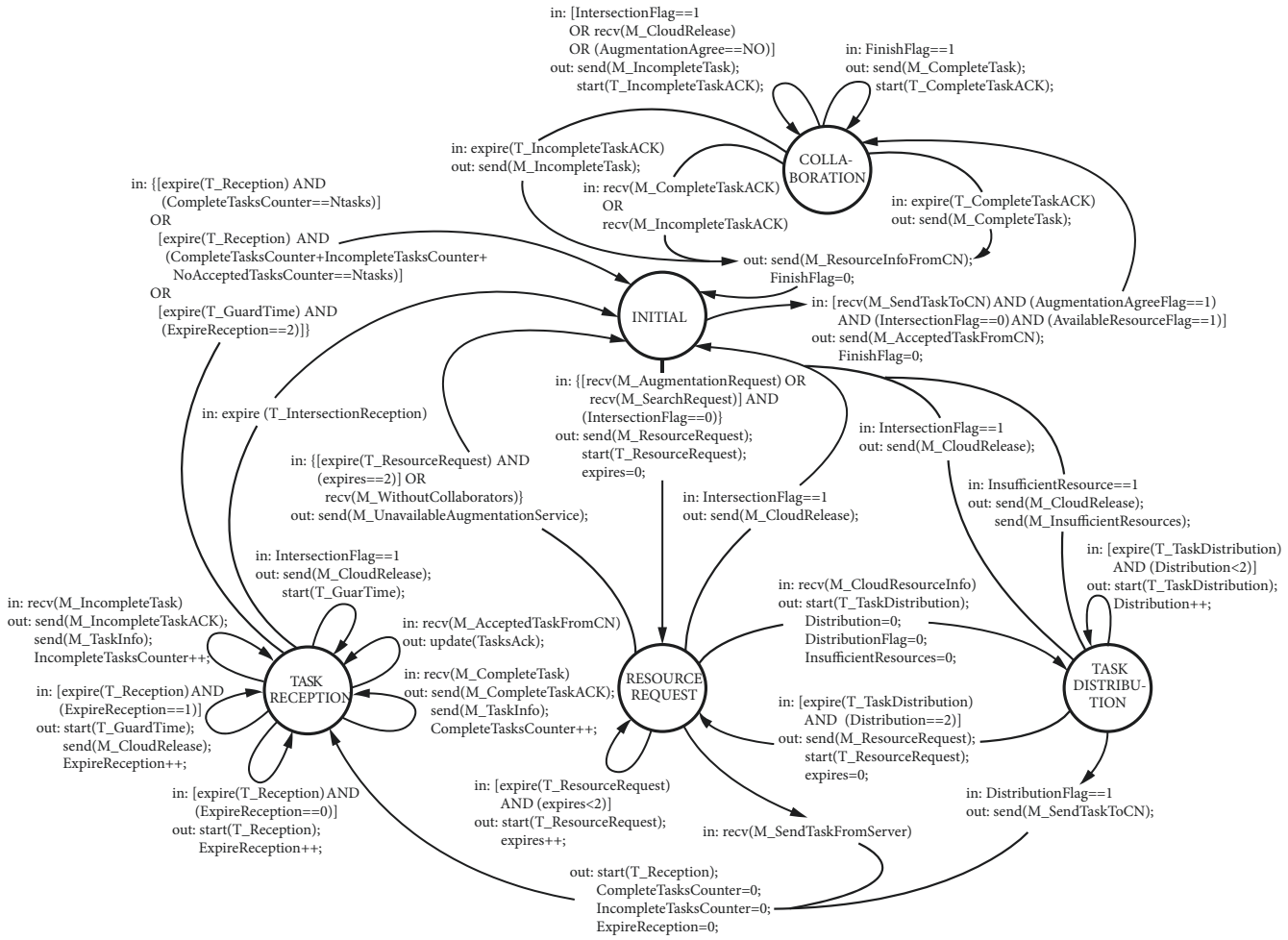

Figure 10: State machine of our S-CMA approach that supports the proposed NaaS model. This figure is reproduced from Ordóñez-Morales, E. F., et al. [20] (under the Creative Commons Attribution License/public domain).

the VANET is crucial to support the virtualization mechanisms on which both the routing protocol and our S-CMA approach work. Specifically, the nodes need to communicate to each other, forward packets, receive messages, download information, etc. For that reason, in this section we describe the mechanism we have envisaged to encourage the nodes to collaborate with each other at the different levels involved in the proposal: virtualization, routing, and sporadic Cloud-based Mobile Augmentation (recall Figure 1).

According to the review presented in Section 2.2, we have discarded a credit-based solution and opted for a reputation-based collaboration approach. The credit-based mechanisms are hard to implement because it is not easy to decide when the credit must be paid to the CNs for their augmentation services. If the CN receives the credit before the augmentation, it would be possible that this node rejects finally to lend its resources; if the payment is made after, some malicious CNs could forward any number of packets, not necessary referring to the AppN-requested content [37].

Specifically, in our collaboration mechanism all the (physical) nodes involved in the communications have *reputation values*, which are increased as the nodes collaborate at the different levels of the protocol stack depicted in Figure 1. (Note that our approach starts to work with reputation values that are estimated by averaging historical values, because initially the nodes still have not collaborated (or these values have not reached the server yet).) In particular,

(i) the AppNs and CNs have a reputation value as per the amount of data sent, which are properly received by the destination nodes, at virtualization, routing, and augmentation levels;

(ii) besides, the CNs have a second reputation value as per the amount of individual/popular contents that are downloaded from the remote server when augmenting the AppNs; this second value depends on the size of data downloaded from the server and not on the data that the AppNs do receive (because it would not be fair to limit the reputation of the CNs to the network conditions).

Following the guidelines of the existing approaches, we adopt as manager entity the remote server where the contents are available. The nodes must register in this server, which provides a security layer to ensure that (i) all the nodes have worked properly in the forwarding and downloading of contents and (ii) their reputation values, respectively, have been correspondingly updated. Specifically, each node involved in our communications maintains a table including (i) the MAC addresses of the destination nodes to which data has been forwarded and (ii) their respective reputation values. This information is sent to the server when a connection to the Internet (either via AP or via own 3G/4G connection) is available. After receiving all the contributions, the server proceeds as follows:

(i) First, the server verifies the forwarding and downloading tasks performed by each type of node (AppN or CN), by matching both the amount of data that have been sent by the origin nodes and received by the destination nodes, and the amount of data that have been downloaded (as appropriate).

(ii) Second, the server updates the corresponding reputation scores for AppNs and CNs and sends them to the nodes via PUSH notifications.

The reputation values of the CNs and AppNs take a crucial role. Specifically, we use them to prioritize the augmentation requests performed by several AppNs, so that if many of these nodes need to be augmented and there are enough augmentation resources, the CNs' bandwidth is lent first to the AppNs with the highest reputation values. To this aim, the information related to the reputation of each node is also communicated to the L1VNs (located at the intersections) and maintained as PSI after the *resource discovery* process described in Section 4.

The reputation scores of the CNs are used for dimensioning rightly the augmentation (sporadic) cloud, because an excessive ratio of CNs can affect negatively the performance of the ad hoc network (due to overhead, collisions, and packet losses, as we will detail in Section 5). This way, in case of a surplus of collaborators, our S-CMA approach can select CNs by considering not only the average time they will stay in a road segment and their available bandwidth (as shown in (2)), but also their respective reputation scores. Besides, each node can work as CN or as AppN or even take both roles. For that reason, the reputation values of a CN can be exploited when this node needs to be augmented (turning into a AppN). According to what commented above, in this scenario, the higher the reputation score this node has reached as CN is, the most preferential its augmentation request will be as AppN.

Lastly, the CNs can be also incentivized by considering our S-CMA approach as the only opportunity to enjoy context-aware tailor-made applications that will not be available outside of our opportunistically deployed sporadic network (where membership is conditioned to the acceptance of sharing some own resources in exchange for using other external ones, if necessary).

5. Experimental Evaluation

In order to validate our S-CMA approach, we propose a scenario of simulation that considers the center of the city of Cuenca in Ecuador (previously captured in OpenStreetMap), covering an area of 1375x1375 meters with 7x7 two-way streets of the city center. Five APs are located within the area of simulation as depicted in Figure 11, which have a coverage area of 50 meters that guarantees connectivity to the remote server where the AppNs-requested contents are available.

The vehicles are equipped with on-board units including GPS, 802.11p, Wi-Fi, and 3G/4G connections. To simulate the communications we use: (i) ns-3 simulator with IEEE 802.11p PHY/MAC (adopting the shadowing radio propagation model [48]) and all the protocols on top, and (ii) SUMO to get realistic mobility traces for 300 vehicles on the streets of the simulation area. Besides, the following settings have been considered in our experiments:

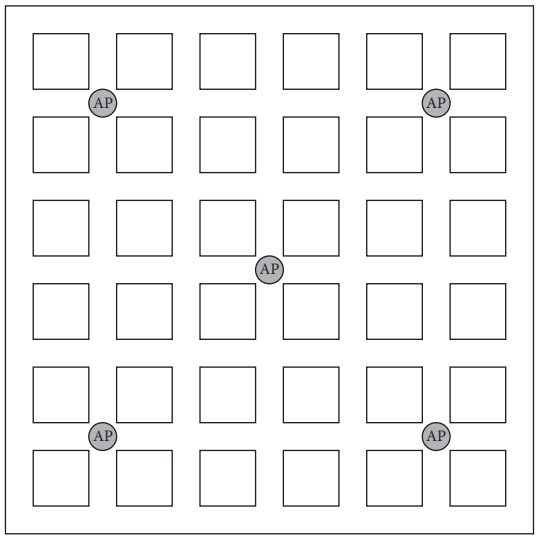

Figure 11: Distribution of the 5 APs within our simulation scenario.

(1) The CNs can handle several augmentation requests (by lending resources to multiple AppNs simultaneously), by resorting first to near APs and, in case of unavailability of them, to their own 3G/4G connections.

(2) The AppNs can just download contents from the Internet through either near APs or collaborator nodes willing to lend their 3G/4G connections.

(3) The AppNs request 5 MB of individual content and 5 MB of popular contents.

(4) The servers do not use compression.

(5) 10% of total nodes are AppNs that request augmentation processes to download popular/individual contents.

(6) We consider diverse proportions of collaborator nodes within the VANET (specifically, 25/50/75% of CNs) with the aim of representing pessimist/middle/optimistic collaboration environments.

(7) The timers used in our augmentation approach (see Table 2) have been settled to 1ms.

(8) Lastly, we measure the performance of our S-CMA approach in terms of *average number of downloads*, *average download time*, and *S-CMA overhead* against the percentage of CNs within the VANET (25%, 50%, and 75% of CNs), comparing the two collaborative download models described in the paper (ICD or individual content download vs PCD or popular content download).

We start analyzing the average amount of downloads carried out by the CNs in the two experimented download models.

As shown in Figure 12, the average number of contents downloaded by the CNs is greater in PCD than in ICD. The reason behind these figures is related to the nature of

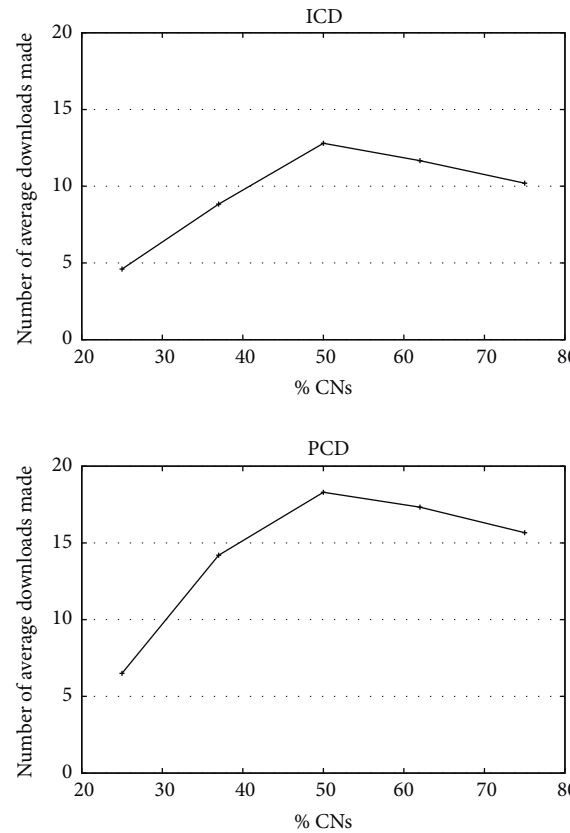

Figure 12: Average number of downloads performed in ICD and PCD against different ratios of CNs within the VANET.

the AppN-requested contents: while the individual contents handled in ICD must be always retrieved from the remote server, in PCD some chunks of popular contents are already stored in the CNs, and therefore these collaborators can deliver very fast these pieces to the AppN, thus being ready again for performing new downloads. This behavior leads to an increase of average number of downloads as the number of available CNs gets higher. However, this trend is reversed for proportions greater than 50% of CNs, since having more CNs in the road segment makes it easier to form sporadic clouds during the augmentation process. A high number of augmentation requests cause greater occupation of the transmission medium, resulting in collisions and packet losses that degrade the performance of the VANET and reduce the average total number of downloads made by the CNs for their respective AppNs.

The evolution of the average number of augmentation requests and the amount of formed sporadic clouds in both download models are depicted in Figures 13 and 14, respectively.

As explained in Section 4, after forming the sporadic cloud in the road segment where the AppN is located, this node waits until the CNs deliver the requested content. During this period of time, the AppN cannot trigger new augmentation requests, unless its CNs deliver the requested content before the corresponding timer expires or the AppN enters an intersection. In this case, the CNs deliver the chunks

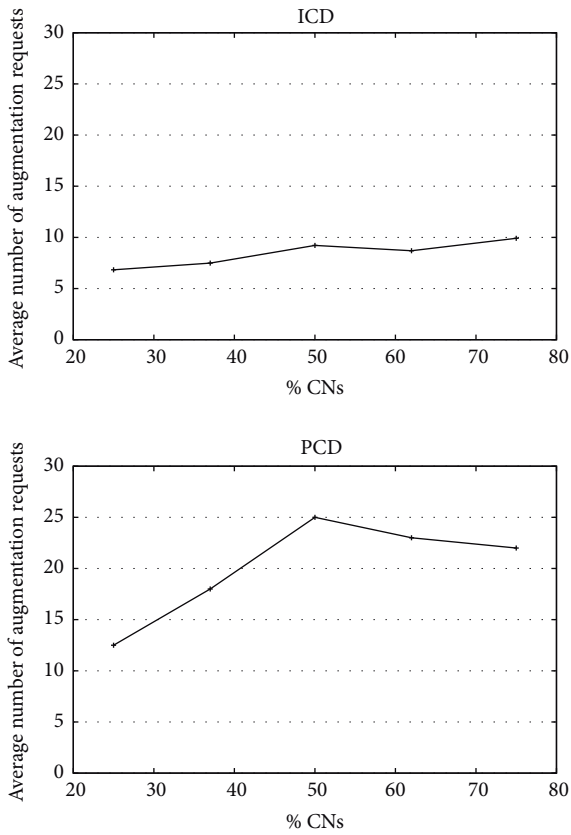

Figure 13: Average number of augmentation requests performed by AppNs in ICD and PCD against different ratios of CNs within the VANET.

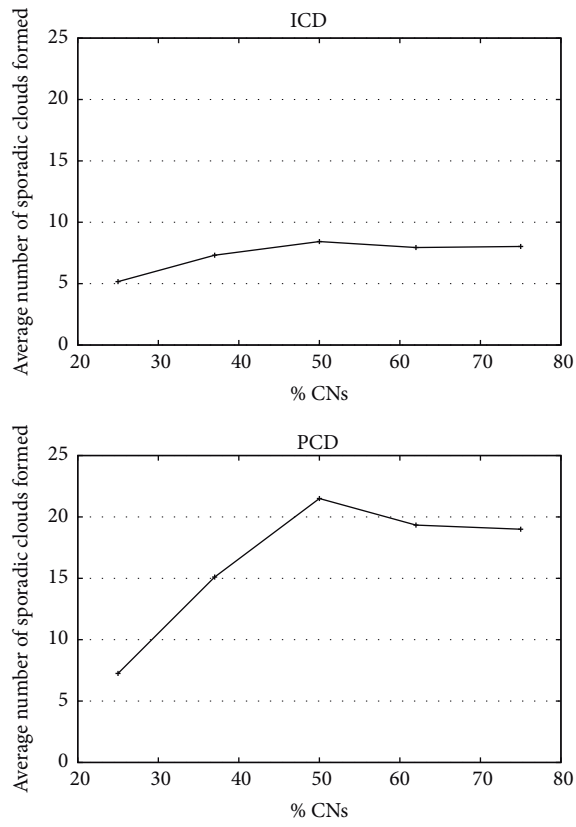

Figure 14: Average number of augmentation sporadic clouds formed in ICD and PCD against different ratios of CNs within the VANET.

of the content available until that moment. Next, a new process of augmentation is triggered in the road segment followed by the AppN after leaving the intersection (in a totally seamless handover for the applications), with the goal of getting the missing pieces of the content.

As shown in Figure 13, the number of augmentation requests dispatched on average is lower in ICD than in PCD. As mentioned before, in ICD the CNs must download all the chunks (for being content of interest just for a particular user in the VANET) and therefore dispatching each request takes longer. Instead, in PCD, as many popular chunks may already be available in the CNs, the collaborators can deliver directly their contents to the AppN, thus remaining free to attend new augmentation requests. In addition, since there are fewer augmentation requests, fewer sporadic clouds in each road segment (see Figure 14) are also formed in ICD than in PCD, where both parameters increase proportionally with the number of CNs available. As shown in Figures 13 and 14, both values decrease for greater ratios of CNs due to a higher occupation of the transmission medium and the consequent frequent collisions and packet loss aforementioned.

The explanations given above justify also the results achieved for average download times and S-CMA overhead.

(i) As depicted in Figure 15, in both models the download time of each 5MB content gets lower as the number of CNs willing to lend their bandwidths increases, reaching the minimum value with a ratio of 50% of CNs (when this parameters remain almost stable). The benefits of PCD due to the availability of many popular chunks in the CNs are noticeable in our simulations, as well as the degradation suffered in the download time due to a transmission medium congested for a higher number of augmentation requests and sporadic clouds formed.

(ii) As shown in Figure 16, the measured S-CMA overhead is greater in PCD than in ICD due to the fact that the amount of control information that must be exchanged among CNs and AppNs (to orchestrate the augmentation process) is also higher. The figures depicted in both charts also confirm the congestion of the VANET for a proportion of CNs greater than about 60%. In particular, when the VANET performance is degraded, the control messages coming from the S-CMA layer are lost more frequently and must be sent again, which increases the networking overhead.

To conclude our experimental design, we depict in Figure 17 the average download speeds achieved in PCD and ICD for the AppNs involved in our simulations, whose values have been compared considering the typical/real download rates for 3.5G (4 Mbps), 4G (15 Mbps), and 802.11p (6 Mbps),

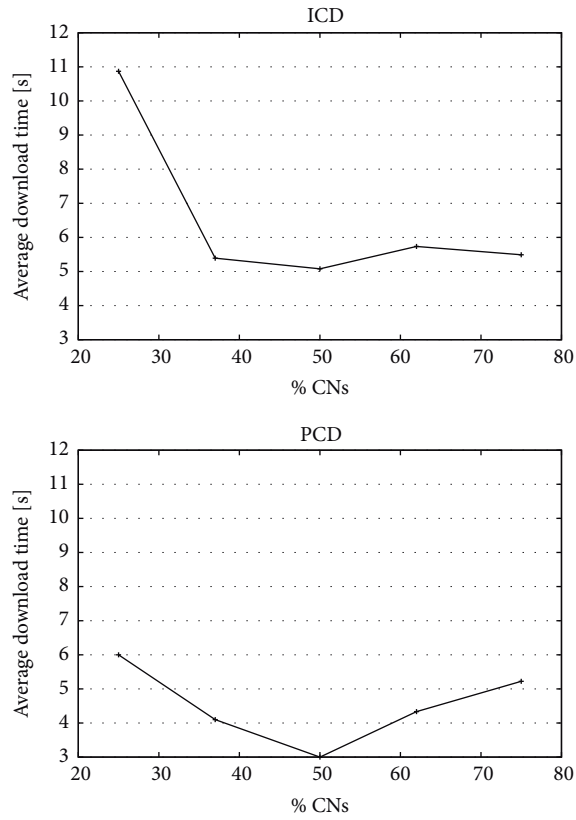

Figure 15: Average download time in ICD and PCD against different ratios of CNs within the VANET.

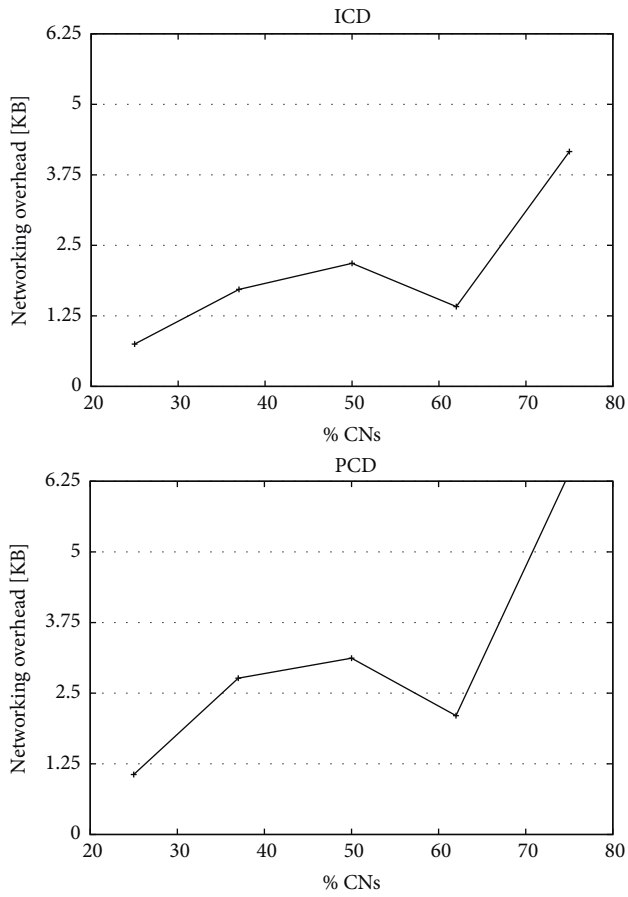

Figure 16: Average networking overhead measured in ICD and PCD against different ratios of CNs within the VANET.

which are the connections that our CNs can use during the collaborative download of web contents. These values have been taken from http://kenstechtips.com/index.php/download-speeds-2g-3g-and-4g-actual-meaning. In particular, the typical download rate for 802.11p networks is about 11 Mbps. We are considering a real value of 6 Mbps as reference for the WiFi APs located at the intersections, assuming that the presence of buildings degrades network performance. As expected, PCD outperforms ICD for the reasons given throughout this section, achieving download speeds that are (i) much better than the rates of 3.5G and WiFi connections via 802.11p (13.33 Mbps vs 4 Mbps and 6 Mbps, respectively) and (ii) slightly lower than the rates of 4G connections (13.33 Mbps vs 15 Mbps). This last point is due to the fact that the congestions, interferences, and the movement of CNs during the augmentation process reduce the real download speed, thus shaping a nonideal scenario where the rate of 15Mbps of a 4G connection is not feasible/realistic. Anyway, the achieved results are certainly promising for a CMA approach like ours, where, unlike existing related works, the execution of mobile applications does not suffer disruption as AppN and CNs move away, thus switching seamlessly between different augmentation clouds.

6. Conclusions and Further Work

In this paper we have presented an approach to CMA that enables execution of intensive-resource applications in a vehicle (application node) within a VANET, by deploying a NaaS model that borrows augmentation resources. The augmentation resources gain advantages from two sources. The first is constituted by the sporadic clouds of close (collaborator nodes) vehicles that move along a road segment during a period of time, while the second source is the roadside-fixed unit. Both sources of augmentation maintain connections to a central server that manages the content desired by the application node. The service model enables individualized download of web contents (ICD) and access to popular contents that are discovered within the VANET and delivered to the interested vehicles (PCD). Unlike existing CMA solutions, our S-CMA approach mitigates to a large extent the problems related to latency, excessive bandwidth consumption, and constraints related to relative locations between the vehicles that require augmentation and the ones that lend resources. As major novelty, our approach is the first proposal that explores synergies between CMA solutions and virtualization mechanisms, by handling stationary virtual nodes instead of on-move vehicles. The notion of virtual node has been also exploited when routing traffic by a robust and efficient protocol working in tandem with the virtualization layer. Our virtualization mechanisms manage the mobility of the cluster of collaborator nodes, an issue that has not been yet completely resolved in the literature

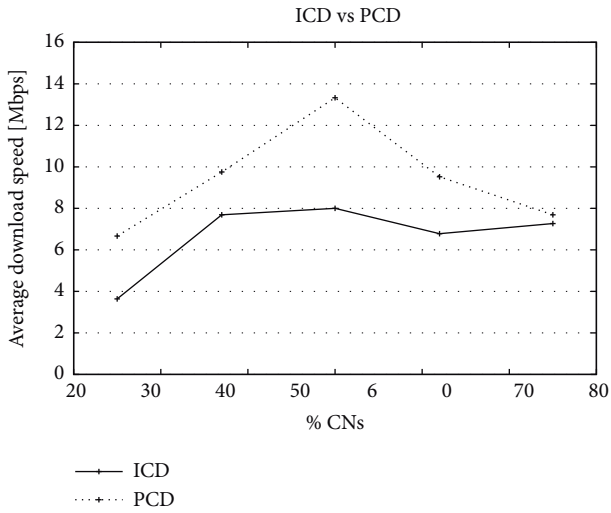

Figure 17: Average download speeds achieved for the augmented AppNs (in ICD and PCD) against different ratios of CNs within the VANET.

where the most sophisticated solutions interrupt the augmentation service once AppN and CNs have moved away. Instead, our virtualization layer enables stable and reliable communication environments even in scenarios of high mobility where the (application and collaborator) vehicles move freely, and where disruptions in the augmentation service are automatically restored (from the S-CMA layer) when AppN and/or CNs leave the current sporadic cloud as they separate.

We have evaluated the performance of our approach, achieving very promising results in terms of number of downloads and average download speed (for AppNs), average download time (of CNs), and networking overhead. These results showed that PCD outperformed ICD, where the AppNs have accessed a greater amount of web contents in less time, at the cost of slightly increasing the overhead in the VANET. This reveals the benefits of the gossip mechanisms adopted in our approach in order to discover the availability of popular chunks in some CNs and deliver them soon to the corresponding AppNs. Besides, our simulations have also allowed us to detect that the number of CNs within the augmentation sporadic clouds must be properly dimensioned in order to prevent excessively occupied transmission mediums, high overhead, frequent collisions, and consequent packet losses. In this dimensioning procedure, our approach exploits a reputation-based mechanism that enables not only identifying the best CNs (in case of surplus of collaborators) but also prioritizing the augmentation requests made by the AppNs (when there are no enough augmentation resources).

As further work, we plan to compare our approach with existing CMA solutions in order to quantify the improvements of our refinements at virtualization/routing/mobile augmentation-level with regard to solutions that suffer restrictions related to the mobility of the augmentation cluster. Besides, we are working on the deployment of new "X"aaS models (specifically, SEnsing and SToring as a Service models) on the foundations of our S-CMA paradigm, with the goal of enriching the mobile users' experience in vehicular environments.

Data Availability

The simulation data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been supported by the European Regional Development Fund (ERDF) and the Galician Regional Government under agreement for funding the Atlantic Research Center for Information and Communication Technologies (AtlantTIC), by the Program for the Consolidation and Structuring of Competitive Research Groups within the Galician R&D&I System, by the Ministerio de Educación y Ciencia (Gobierno de España) research project TIN2017-87604-R. The authors are also thankful to FREEPIK for providing us with some of the graphics used in this document.

References

[1] M. Gerla, C. Wu, G. Pau, and X. Zhu, "Content distribution in VANETs," *Vehicular Communications*, vol. 1, no. 1, pp. 3–12, 2014.

[2] S. Jia, Z. Liu, K. Zhu, L. Zhang, Z. M. Fadlullah, and N. Kato, "Bus-Ads: Bus-based priced advertising in VANETs using coalition formation game," in *Proceedings of the IEEE International Conference on Communications, ICC 2015*, pp. 3628–3633, London, UK, June 2015.

[3] G. S. Khekare, "Design of emergency system for intelligent traffic system using VANET," in *Proceedings of the 2014 International Conference on Information Communication and Embedded Systems, ICICES 2014*, Chennai, India, February 2014.

[4] B. Liu, D. Jia, J. Wang, K. Lu, and L. Wu, "Cloud-assisted safety message dissemination in VANET-cellular heterogeneous wireless network," *IEEE Systems Journal*, vol. 11, no. 1, pp. 128–139, 2017.

[5] H. S. Hassanein, S. Abdelhamid, and K. Elgazzar, "A framework for vehicular cloud computing," in *Proceedings of the International Conference on Connected Vehicles and Expo, ICCVE 2015*, pp. 238-239, Shenzhen, China, October 2015.

[6] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: motivation, taxonomies, and open challenges," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 337–368, 2014.

[7] M. Gerla, "Vehicular cloud computing," in *Proceedings of the 11th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2012)*, pp. 152–155, Ayia Napa, Cyprus, June 2012.

[8] R. Yu, Y. Zhang, S. Gjessing, W. Xia, and K. Yang, "Toward Cloud-based vehicular networks with efficient resource management," *IEEE Network*, vol. 27, no. 5, pp. 48–55, 2013.

[9] R. Karim, "VANET: superior system for content distribution in vehicular network applications," Tech. Rep., Rutgers University, Department of Computer Science, 2008.

[10] S. Hussain, Z. Hamid, and N. S. Khattak, "Mobility management challenges and issues in 4G heterogeneous networks," in *Proceedings of the First International Conference on Integrated Internet Ad-hoc and Sensor Networks (InterSense)*, p. 14, Nice, France, May 2006.

[11] C. Wang, Y. Li, D. Jin, and S. Chen, "On the serviceability of mobile vehicular cloudlets in a large-scale urban environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2960–2970, 2016.

[12] S. Olariu, I. Khalil, and M. Abuelela, "Taking VANET to the clouds," *International Journal of Pervasive Computing and Communications*, vol. 7, no. 1, pp. 7–21, 2011.

[13] E. Lee, E.-K. Lee, M. Gerla, and S. Y. Oh, "Vehicular cloud networking: architecture and design principles," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 148–155, 2014.

[14] S. Abolfazli, Z. Sanaei, M. Shiraz, and A. Gani, "MOMCC: market-oriented architecture for mobile cloud computing based on service oriented architecture," in *Proceedings of the 2012 1st IEEE International Conference on Communications in China Workshops, ICCC 2012*, pp. 8–13, Beijing, China, August 2012.

[15] K. Xu, K.-C. Wang, R. Amin, J. Martin, and R. Izard, "A fast cloud-based network selection scheme using coalition formation games in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 11, pp. 5327–5339, 2015.

[16] M. H. Firooz and S. Roy, "Collaborative downloading in VANET using network coding," in *Proceedings of the IEEE International Conference on Communications (ICC '12)*, pp. 4584–4588, Ottawa, ON, Canada, June 2012.

[17] N. Kumar, J. J. Rodrigues, and N. Chilamkurti, "Bayesian coalition game as-A-service for content distribution in internet of vehicles," *IEEE Internet of Things Journal*, vol. 1, no. 6, pp. 544–555, 2014.

[18] H. R. Arkian, R. E. Atani, A. Diyanat, and A. Pourkhalili, "A cluster-based vehicular cloud architecture with learning-based resource management," *The Journal of Supercomputing*, vol. 71, no. 4, pp. 1401–1426, 2015.

[19] E. F. Ordonez-Morales, Y. Blanco-Fernandez, J. F. Bravo-torres, M. Lopez-Nores, V. Saians-Vazquez, and J. J. Pazos-arias, "S-CMA: sporadic cloud-based mobile augmentation supported by an ad-hoc cluster of moving handheld devices and a virtualization layer," in *Proceedings of the 2015 Fifth International Conference on Innovative Computing Technology (INTECH)*, pp. 152–157, Vigo, Spain, May 2015.

[20] E. F. Ordonez-Morales, J. F. Bravo-Torres, Y. Blanco-Fernandez, M. Lopez-Nores, V. Saians-Vazquez, and J. J. Arias, "Exploiting virtualization and sporadic clouds for collaborative downloading in vanets: a new networking as a service model," in *Proceedings of the 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*, pp. 50–57, Vienna, Austria, August 2016.

[21] J. F. Bravo-Torres, M. Lopez-Nores, Y. Blanco-Fernandez, J. J. Pazos-Arias, M. Ramos-Cabrer, and A. Gil-Solla, "Optimizing reactive routing over virtual nodes in VANETs," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2274–2294, 2016.

[22] S. Ahmed and S. S. Kanhere, "VANETCODE: network coding to enhance cooperative downloading in vehicular ad-hoc networks," in *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC '06)*, pp. 527–532, ACM, Vancouver, BC, Canada, July 2006.

[23] K. C. Lee, S.-H. Lee, R. Cheung, U. Lee, and M. Gerla, "First experience with CarTorrent in a real vehicular ad hoc network testbed," in *Proceedings of the Mobile Networking for Vehicular Environments (MOVE '07)*, pp. 109–114, Anchorage, AK, USA, May 2007.

[24] U. Lee, J.-S. Park, J. Yeh, G. Pau, and M. Gerla, "Code torrent: content distribution using network coding in VANET," in *Proceedings of the 1st International Workshop on Decentralized Resource Sharing in Mobile Computing and Networking (MobiShare '06)*, pp. 1–5, ACM, Los Angeles, CA, USA, September 2006.

[25] B. B. Chen and M. C. Chan, "MobTorrent: a framework for mobile internet access from vehicles," in *Proceedings of the 28th Conference on Computer Communications (INFOCOM)*, pp. 1404–1412, Rio De Janeiro, Brazil, April 2009.

[26] N. Liu, M. Liu, G. Chen, and J. Cao, "The sharing at roadside: vehicular content distribution using parked vehicles," in *Proceedings of the 31st Annual IEEE International Conference on Computer Communications (INFOCOM)*, pp. 2641–2645, Orlando, FL, USA, March 2012.

[27] S. Arif, S. Olariu, J. Wang, G. Yan, W. Yang, and I. Khalil, "Datacenter at the airport: Reasoning about time-dependent parking lot occupancy," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 11, pp. 2067–2080, 2012.

[28] M. Eltoweissy, S. Olariu, and M. Younis, "Towards autonomous vehicular clouds," in *Proceedings of the Ad Hoc Networks*, vol. 49, pp. 1–16, Springer, 2010.

[29] S. Olariu, T. Hristov, and G. Yan, "The next paradigm shift: from vehicular networks to vehicular clouds," in *Mobile Ad Hoc Networking: The Cutting Edge Directions*, pp. 645–700, Wiley and Sons, 2012.

[30] W. Huang and L. Wang, "ECDS: Efficient collaborative downloading scheme for popular content distribution in urban vehicular networks," *Computer Networks*, vol. 101, pp. 90–103, 2016.

[31] A. Nandan, S. Das, G. Pau, M. Gerla, and M. Y. Sanadidi, "Cooperative downloading in vehicular ad-hoc wireless networks," in *Proceedings of the 2nd Annual International Conference on Wireless On-Demand Network Systems and Services (WONS '05)*, pp. 32–41, January 2005.

[32] C. Huang, C. Yang, and H. Lin, "A bandwidth aggregation scheme for member-based cooperative networking over the hybrid VANET," in *Proceedings of the 2011 IEEE 17th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 436–443, Tainan, Taiwan, December 2011.

[33] T. Wang, L. Song, and Z. Han, "Collaborative data dissemination in cognitive VANETs with sensing-throughput tradeoff," in *Proceedings of the 2012 1st IEEE International Conference on Communications in China, ICCC 2012*, pp. 41–45, Beijing, China, August 2012.

[34] T. Chen, L. Zhu, F. Wu, and S. Zhong, "Stimulating cooperation in vehicular ad hoc networks: a coalitional game theoretic approach," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 2, pp. 566–579, 2011.

[35] S.-B. Lee, J.-S. Park, M. Gerla, and S. Lu, "Secure incentives for commercial ad dissemination in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 6, pp. 2715–2728, 2012.

[36] G. Ananthanarayanan, V. N. Padmanabhan, C. A. Thekkath, and L. Ravindranath, "Collaborative downloading for multi-homed wireless devices," in *Proceedings of the 8th IEEE Workshop on Mobile Computing Systems and Applications, HOTMOBILE 2007*, pp. 79–84, Tucson, AZ, USA, February 2007.

[37] P. Caballero-Gil, J. Molina-Gil, C. Hernandez-Goya, and C. Caballero-Gil, "Stimulating cooperation in self-organized vehicular networks," in *Proceedings of the 2009 15th Asia-Pacific Conference on Communications, APCC 2009*, pp. 346–349, Shanghai, China, October 2009.

[38] F. Dötzer, L. Fischer, and P. Magiera, "VARS: a vehicle ad-hoc network reputation system," in *Proceedings of the 6th IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, WoWMoM '05*, pp. 454–456, Taormina-Giardini Naxos, Italy, June 2005.

[39] Q. Li, A. Malip, K. M. Martin, S.-L. Ng, and J. Zhang, "A reputation-based announcement scheme for VANETs," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 9, pp. 4095–4108, 2012.

[40] J. F. Bravo-Torres, M. López-Nores, Y. Blanco-Fernández, J. J. Pazos-Arias, and E. F. Ordóñez-Morales, "VaNetLayer: A virtualization layer supporting access to web contents from within vehicular networks," *Journal of Computational Science*, vol. 11, pp. 185–195, 2015.

[41] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector (AODV) routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, pp. 90–100, New Orleans, La, USA, February 1999.

[42] M. Jerbi, S.-M. Senouci, T. Rasheed, and Y. Ghamri-Doudane, "Towards efficient geographic routing in urban vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 9, pp. 5048–5059, 2009.

[43] J. F. Bravo-Torres, M. Lopez-Nores, J. V. Saians-Vazquez, Y. Blanco-Fernandez, and J. J. Pazos-Arias, "An efficient combination of topological and geographical routing for VANETs on top of a virtualization layer," in *Proceedings of the 2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pp. 1–5, Glasgow, United Kingdom, May 2015.

[44] E. F. Ordonez-Morales, V. Saians-Vazquez, J. F. Bravo-Torres, Y. Blanco-Fenandez, and M. Lopez-Noresi, "Leveraging proactive and reactive intersection-based routing protocols for collaborative downloading in VANETs," in *Proceedings of the 2016 8th IEEE Latin-American Conference on Communications (LATINCOM)*, pp. 1–6, Medellin, Colombia, November 2016.

[45] Y. Wiseman, "Can flight data recorder memory be stored on the cloud?" *Journal of Aviation Technology and Engineering*, vol. 6, no. 1, p. 3, 2016.

[46] Y. Wiseman, "Unlimited and protected memory for flight data recorders," *Aircraft Engineering and Aerospace Technology*, vol. 88, no. 6, pp. 866–872, 2016.

[47] A. Baiocchi and F. Cuomo, "Infotainment services based on push-mode dissemination in an integrated VANET and 3G architecture," *Journal of Communications and Networks*, vol. 15, no. 2, pp. 179–190, 2013.

[48] P. K. Singh and K. Lego, "Comparative study of radio propagation and mobility models in vehicular adhoc network," *International Journal of Computer Applications*, vol. 16, no. 8, pp. 37–42, 2011.