# SMART: An Efficient Resource Search and Management Scheme for Vehicular Cloud-connected System

Rodolfo I. Meneguette[‡,†], Azzedine Boukerche [†], Robson de Grande [†]
[‡]Federal Institute of São Paulo (IFSP), Catanduva, São Paulo, Brazil
[†]School of Electrical Engineering and Computer Science, University of Ottawa - Canada
Email: meneguette@ifsp.edu.br, {boukerch, rdegrande}@site.uottawa.com

***Abstract*—A Vehicular Cloud generally focuses on several aspects, which include providing a set of computational services at low cost to vehicle drivers; minimizing traffic congestion, accidents, travel time, and environmental pollution; and ensuring the use of low energy and real-time services of software, platforms, and infrastructure with QoS to drivers. The largest challenge in vehicular mobile Clouds consists of creating a mechanism for management and search resources that does not depend on roadside infrastructure, which consequently requires the spontaneous and dynamic creation of a Cloud through the resources shared by vehicles. Therefore, the system must enable collaboration and co-operation between vehicles so that they may establish connections to provide resources. To address this challenge, we propose a peer-to-peer protocol to assist in the discovery and management of resources in a vehicular mobile Cloud without depending on the support of a roadside infrastructure so that vehicles are expected to organize themselves and establish collaborations to manage and share their resources. Simulation results show that the proposed approach introduces a short search time of approximately 0.5 (ms) to seek resources in one hop and approximately 0.9 (ms) to seek more of a hop in resources. Furthermore, the proposed protocol enables a high availability of resources, about 87%.**

## I. Introduction

The Vehicular Cloud represents a paradigm shift in technology, which takes advantage of Cloud Computing to serve those drivers that make use of the vehicular network [1], [2]. The objectives of the Vehicular Cloud are to provide several computational services at a low cost to drivers; to minimize traffic congestion, accidents, travel time, and environmental pollution; and to ensure the use of low energy and real-time services of software, platforms, and infrastructure with QoS restrictions to drivers [3]–[5].

Recently, a few research projects have been reported working on the combination of Cloud Computing and vehicular networks. In [6], [7], the concept of autonomous Vehicular Clouds (AVCs) was introduced to exploit the underutilized resources in vehicular ad hoc networks (VANETs). A model of a platform as a service (PaaS) was designed in [8] to support Cloud services for mobile vehicles. The work in [9] described architectures of Vehicular Clouds (VCs), vehicles using Clouds (VUCs), and hybrid Clouds (HCs). Vehicles act as Cloud service providers in VCs and as clients in VUCs, they act as both in HCs. Nevertheless, either most of these works require certain aid from the roadside infrastructure to support the sharing of and search for resources in the Cloud, or they do not consider that the vehicles may have computational resources that can be shared by other vehicles. Therefore, the greatest challenge in the Vehicular Cloud consists of creating a mechanism for managing and identifying resources, which does not necessarily depend on the roadside infrastructure. The mechanism is designed to setup a Cloud spontaneously through the resources shared by vehicles. Therefore, such a system must consider the collaboration and co-operation among vehicles so that they may establish connections on-demand and dynamically to make their resources available. We contemplate this system in our paper by naming it vehicular mobile Cloud.

With this challenge in mind, we propose a new peer-to-peer protocol for search and management of resources and services in the vehicular mobile Cloud. The protocol does not require the assistance of any external, infrastructural road element, and it is based entirely on resource sharing among vehicles. Moreover, vehicle mobility plays a critical role in our protocol since the movement of vehicles restricts the resource service time and delimits the algorithms for search and maintenance, which must consider the brevity in which connections are established and ceased. The proposed protocol allows collaboration and co-operation between vehicles through enforcing that they are capable of sharing their resources over the vehicular network and can identify the hosts of sought services.

The remainder of this paper is structured as follows. In Section II, we describe the most relevant related solutions. In Section III, we present our proposed peer-to-peer protocol for search and management in the vehicular mobile Cloud. In Section IV, we introduce our experimental scenario and analyze the simulation results. Finally, in Section V we summarize the work, providing direction for future works.

## II. Related Works

Even through Vehicular Cloud Computing introduces a novel pragmatic paradigm, it has been already explored in certain works, such as the study conducted by Olariu et al. [10], which proposed a new concept called Vehicular Cloud (VC). This particular VC employs underutilized vehicle resources to form a Cloud by aggregating available vehicular computing power. It is worth noting that the proposed system does not

This work is supported by the grants #2015/11536-4 and #2015/18898-9, São Paulo Research Foundation (FAPESP).

offer any advantages to the capabilities of a conventional Cloud because this VC approach is based only on vehicle resources.

Mershed and Artil [11] introduced a mechanism that allowed vehicles in a network to obtain information about necessary services from mobile Cloud servers that are nearby. This work consists of a system called CROWN, which depends on RSUs that act as interfaces and Cloud directories. The system requires that RSUs provide the recorded data to allow damaged vehicles to access to necessary services in the Cloud while they are within the RSU coverage area. The CROWN system failed to consider the on-board computer as a Cloud Computing entity that could be available to end users.

Hussain et al. [9] designed a Cloud with three components: Vehicular Clouds (VCs), vehicles using Clouds (VUCs), and hybrid Clouds (HCs). The VC is divided into two categories: a static Cloud, which refers to stationary vehicles and provides services to the Cloud, and a dynamic Cloud, which is configured through an on-demand ad hoc network. The VUC allows a vehicle to connect to the Cloud using traditional RSUs while the HC is a combination of VC and VUC. It should be noted that this solution requires the vehicles in the VC to be still, not in movement. Furthermore, the vehicles can only interact with the traditional Cloud, acting as a gateway. As a major drawback, the vehicle cannot be connected to the MSW if the VC is not available.

Sibai el al. [12], [13] presented a connectivity-aware service provision mechanism in which the service provider vehicle is selected on the basis of several parameters, such as the availability of the requested service and the mobility of the vehicles. This mechanism allows a service requester vehicle to form its own Cloud. The services are exchanged between the service requester vehicle and the service provider vehicles. This matching process is performed by having the requester sending a request message via broadcast to its neighbors. Although the authors create a relay mechanism, this mechanism is unable to troubleshoot the broadcast storm, and thus the solution causes an overhead on the network.

Our proposed solution differs from the others and does not depend on external infrastructure; consequently, our solution establishes communication only between the vehicles. Therefore, we consider that the vehicles have resources that can be shared, thus establishing a vehicular mobile Cloud. Moreover, we propose a management and search mechanism capable of reducing network overhead.

## III. SMART: An Efficient Resource Search and Management Scheme for Vehicular Cloud-connected System

The resource search and management problem can be formulated as follows. Consider a $S$ set of resources that is available in the vehicular network by $V$ vehicles and/or a central Cloud. Vehicles can be connected with a $R$ roadside unity, and each roadside unity can support $RC$ resources. Thus, the objective is to maximize the resource availability and maximize the resource utilization time by the vehicles. Therefore, we propose a solution called SMARt (Search and MAnagement Resource proTocol), that aims to assist in the management and resource search of a vehicular mobile Cloud without the support of a roadside infrastructure. As a result, vehicles must be self-organized and collaborate with other vehicles so that they can manage and share their resources. SMARt is a peer-to-peer protocol based on Gnutella concepts and is designed to create an overlay to facilitate resource management in the vehicles. Figure 1 describes the architecture of the SMARt protocol, which is divided into two major components: (i) resource management, and (ii) routing.

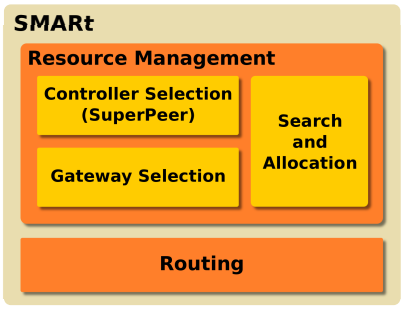


Fig. 1. The General Architecture of the proposed SMARt.

### A. Resource Management

Resource Management is responsible for creating and managing the overlay, as well as for allocating and managing the resources being shared with the vehicles of the mobile Cloud. Resource Management is a cross-layer component; however, the main part of this component works in application layer. This component extends the main messages through the basic concepts in Gnutella, as described in Figure 2. The Ping message is extended by incorporating the location of the vehicle that made the request, the resources that must be allocated to the requester, and the identification and location of the Controller (superpeer) and Gateway. The Pong message contains the same fields, but it expands on having the location and the id of the node that sends the ping message, the status of the requested services, the id gateway, and the controller to which the node is connected. The Query message contains the same location as the requester, the identification and location of the gateway and controller, and the required resources. The Query_Hit message already presents the confirmation for the resource request, the identification and location of the sender, and the location and identification of the gateway and the controller to which it is connected. The identification fields and locations of the gateway and controller are essential information for routing components.

In addition to the overlay management messages, the protocol also makes use of 802.11p control messages (Beacons) since this enables to ascertain the values of speed, direction, and position. Moreover, the shared resources provide information about which resources the vehicle can share. This message serves to assist in the maintenance of the next nodes, as well as the node requesting the controller.

### B. Selection Method

SMARt uses the position of the vehicle in relation to the vehicle that is requesting the resource in order to select the controller and gateway nodes.

The controller is the vehicle in the area of communication that is farthest from the vehicle making the request, and which

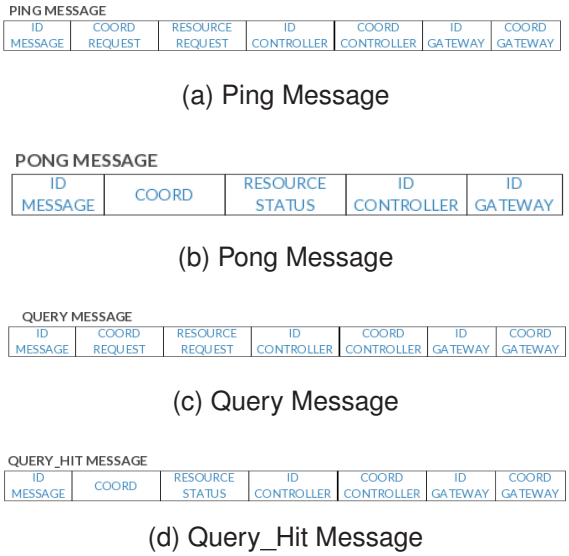


Fig. 2. Structures of the Control Messages.

is traveling in the same direction. This means both vehicles, controller and requester, are on the borders of the coverage area of the communications network. The gateway is comprised of the vehicles that are closest to the requester and are traveling in the same direction.

The information of the controllers and gateways is stored in a database so that the requester can manage the allocation and the search for new resources.

### C. Search and Allocation

When carrying out the allocation procedure and the search for resources, the vehicle is used as a parameter for the location of the nodes, the average speed, the time delay of the messages, and the start time of the service initialization. This information about the vehicles is obtained through the beacons, which the vehicle periodically sends to its neighbors. With the aid of this information, resource allocation can occur if the vehicles are traveling in the same direction. Additionally, the relative velocity between the requester vehicle and the vehicle possessing the resources to be shared is equal to 0; this means that the speed of the vehicles is the same speed or $Distance.req.to.other/Speed.req.to.other > (Delay + Startup.resource)$.

There might be a situation in which there are no vehicles near the requester that meet the above requirements. In this case, the requester sends a message directly to the requested vehicle that is attempting to allocate the resource (B in the diagram of Figure 3). Otherwise, the vehicle sends a request to its controllers, which had become the new requester. When the controller receives the request, it verifies whether there are any vehicles nearby that meet the above requirements. If a vehicle exists, the requester sends a direct message as an attempt to allocate resources from that vehicle and to store the information. If the controller has allocated all the necessary resources, it sends a confirmation response to the requester (C). Otherwise, the controller forwards the request to the next controller, which is either the farthest vehicle towards the front or farthest one from the back, depending on the position of the new requester.

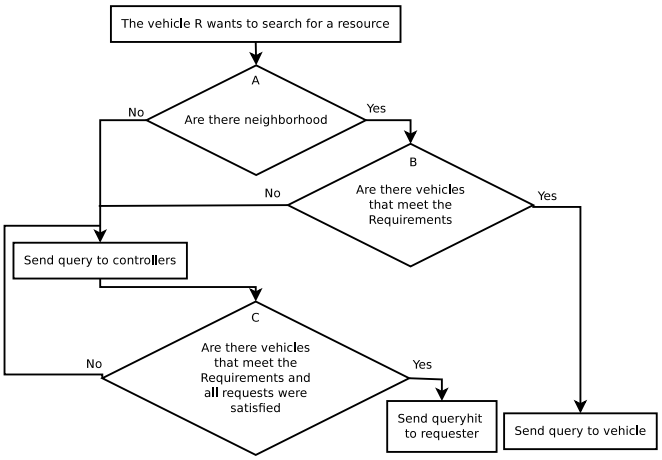


Fig. 3. Selection Methods

### D. Component Operation

When a vehicle requires resources from the vehicular mobile Cloud, the requester checks amongst its neighbors for any vehicle that meets its request (line 1 in algorithm 1). If any vehicle can do so, the requester will store this information and send a query message to the vehicle requesting the resource (lines 2 - 3). Upon receiving the query, the vehicle verifies whether the service is currently idle and responds with a query_hit. When the requester receives a query_hit, it verifies whether the allocation has been made and updates the information in this vehicle. If the allocation has been successful, the service is initiated. Otherwise, the requester selects the controllers (both at the front and back) and sends a query to these controllers (lines 10-11).

**Algorithm 1** Request Resource

```
1: if (check resource available) then
2:     store information
3:     send query
4: end if
5: if (received query_hit) then
6:     check allocation
7:     if (allocate ok) then
8:         update information
9:     else
10:        select Controllers
11:        Send query to Controllers
12:    end if
13: end if
```

When the controllers receive a query, they store the entire information of the requester (line 2 on algorithm 2) and then carry out the task of checking whether any of the vehicles connected to it meets the requirements (line 5). If so, it sends a new query to these vehicles (line 6). Upon receiving the query, it checks whether the vehicle that is currently in service is idle and responds with a query_hit. When controllers receive a query_hit, they store the entire information of the vehicle and send the requester one query_hit including information about the allocation of required resources (lines 10-14). However, if the controller manages to allocate resources to any of the vehicles connected to it, the controller first checks its position in relation to the requester and then retransmit the request to a new controller (lines 15-18). The selected vehicle is then the one at the greatest distance ahead or at the greatest distance behind, depending on its position in elation to the vehicle that

requested the resource.

**Algorithm 2** Controller Management

```
1: if (Receive query) then
2:     store information of message()
3: end if
4: for "each neighbor" do
5:     if (resource available) AND (all resources allocated) then
6:         store information
7:         send query
8:     end if
9: end for
10: if (received query_hit) then
11:     check allocation
12:     if (allocate ok) then
13:         update information
14:         send query_hit to requester
15:     else
16:         select Controllers
17:         send query to Controllers
18:     end if
19: end if
```

After the entire process has been completed, an overlay is created based on the requested resource. Figure 4 illustrates an abstraction of the overlay after a search time. The red dots represent the controller close to the requester, the gray dots are the most distant controllers, the blue dots are the requesters, and the orange dots are the vehicles allocated.

### E. Routing

A Gnutella protocol is a standard used for a reply path of the requested messages, and the same path is sent as a result of a request message. However, this strategy is ineffective in vehicular networks due to their unique features. With this in mind, we recognize the need for a new strategy for the delivery of information. In this work, we have created gateways to assist in this task; the gateway is the bridge between the requester and the controllers, or it is the point between the controllers if any have lost a connection with the requester or the controller.

When a vehicle transmits information, it first verifies that communication is possible with the requester. If so, it sends the data directly to the requester; otherwise, it sends the data to the gateway if the gateway is within reach. If the pre-established gateway is not within the range of the vehicles, it sends the answer to the vehicle that is most distant from its area of communication (in the direction of the requester) until it reaches the gateway or the requester.

## IV. Performance Analysis

This section describes the scenario and performance assessment of the proposed solution. The assessment has been made possible through simulation experiments with the aid of OMNeT++ 4.6[1] network simulator, which is an event-based network simulator. We also used the Simulator for Urban MObility 0.25.0 (SUMO) [14] to build the scenarios and the vehicle mobility model. In addition, the experiments rely on the Veins 4.3 open source network framework[2], which is a well-known tool in the research community that implements the IEEE 802.11p protocol stack and models the signal attenuation caused by obstacles and other features used in our analysis.

[1]http://www.omnetpp.org
[2]http://veins.car2x.org

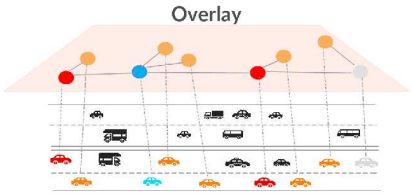


Fig. 4. Selection Methods

### A. Scenario Description

In this evaluation, we employ a realistic scenario for the simulations, which is obtained from OpenStreetMap[3]. The area used comprises a 4.5 km stretch of King's Highway 417, which is a 400-series highway in the Canadian province of Ontario that runs through the city of Ottawa. The highway connects the cities of Montreal and Ottawa and is the backbone of the transportation system in the National Capital Region. Within Ottawa, it forms part of the Queensway.

Three types of vehicles are included in the network to ensure a realistic scenario. The first type consists of vehicles capable of reaching a maximum speed of 120 km/h over a distance of four meters while the second involves vehicles capable of reaching a maximum speed of 90 km/h over a distance of 14 meters. Finally, the third type includes vehicles that are 18 meters long, which are capable of reaching a maximum speed of 90 km/h. This scenario can describe a network consisting of passenger cars, buses, and trucks. It should be noted that the number of vehicles for each type is different during the simulation period. We decided on a network that consists of $50\%$ of cars, $25\%$ of buses, and $25\%$ of heavy trucks. The vehicles can move in both directions on the highway and are positioned at opposite sides, where they move at a constant speed.

Each vehicle is capable of sharing 0 or 2 resources; in other words, some vehicles may have (i) no available resources, (ii) storage and computation resources, (iii) network and computation resources, or (iv) storage and network resources. These resources are selected at random, thus diversifying the types of available opportunities for the set of vehicles to connect.

When the simulation reaches a stable state, the vehicle in the middle of the highway initializes the processes of search and Cloud management by sending the request message (query message), and it looks for the 3 types of shared resources through the vehicle mobile Cloud. This message is either sent to its neighbors or directly to the superpeers (controllers) and thus establishes the entire management infrastructure.

The objective is to evaluate the search time for resources and determine their time of availability, as well as the number of control messages generated in the network that are needed to establish the request and the maintenance of the network.

The proposed solution was compared with a simpler approach that ignores the trajectory of the vehicle and with the solution proposed in [12], named Conaware; the simpler approach was also adopted in previous works [13] for comparison purposes. In this scenario, the following parameters were used: (i) availability of service; (ii) lack of an available service;

[3]http://www.openstreetmap.org/

TABLE I. SIMULATION PARAMETERS

| Parameters | Value |
|---|---|
| Transmission power | 2.2mW |
| Transmission range | 300m |
| Bit rate | 18Mbit/s |
| Highway length | 4.5km |
| Beacons time | 0.5s |
| Ping time | 5s |

(iii) number of generated control messages, disregarding the beacons; (iv) the resource search time for 1 hop; and (v) the time search feature to the n hop.

The bit rate was defined at 18 Mbit/s in the MAC layer and the transmission power at 2.2 mW; thus, the communication range is approximately 300 m by employing a two-ray ground propagation model [15]. Finally, in all obtained results, each point in the curves represents an average of 40 replications, with a confidence interval of $95\%$. Table I shows the configuration and parameters that were used to execute the simulations.

### B. Simulation Results

The number of vehicles was set at 500 vehicles/hour, and the service of the allotted time varied from 10 to 60 seconds with variation of 10 s. This scenario was chosen to make a fair comparison since the compared protocols are used to send their messages via broadcast.

Figure 5a shows the percentages of service availability. It can be noted that SMARt obtained a greater degree of service availability because the proposed protocol did not overload the network with control messages. SMARt reduced the number of collisions on the network and increased availability through the resource allocation mechanism. The allocation mechanism feature is designed to relocate the service that will be lost in other vehicles. The simplistic approach has the worst behavior because in addition to undergoing the broadcast storm, it does not take into account the trajectory of vehicles when allocating resources, leading to a further increase in losses (as seen in Figure 5b). Although Conaware has a retransmission mechanism and selection feature, this protocol also presents overhead and thus causes many collisions between packets.

Figure 5b presents the percentage of lack of services. It can be observed that SMARt achieved a reduction of lack of services of approximately $53\%$ when compared with the simplistic approach and a reduction of $21\%$ when compared with Conaware. When analyzing the point of 60 s to the point with 10 s, our solution provided a reduction of $10\%$ in the lack of services when compared with Conaware and a reduction of $30\%$ to the simplistic approach. This reduction is originated from the fact that SMARt reduces the number of control messages, as well as improving the management of the allocation of resources between the vehicles.

Figure 5c describes the search time for 1 hop, as it is a search feature for the requester's neighbors. The results show that SMARt presents a smaller search time since it had prior knowledge of the available resources, either next to the requester or next to the controller. This prior knowledge allows greater agility in the search for and allocation of resources. With regard to the point of service with 60 s requests, it is clear that the SMARt led to a reduction of search time of about $50\%$ and $55\%$ when compared with Conaware and the simplistic approach, respectively.

Figure 5d displays the results of the search time for the n hop. When looking for more distant node resources, SMARt performed a lower search time because it reduced network overhead besides containing drivers' knowledge of available resources in the surrounding area. Analyzing both Conaware and the simplistic approach results, it can be noted that the difference between them is negligible. This little difference is due to the fact that both approaches communicate via broadcast, which causes a lot of collisions on the network and makes it difficult to use the research feature to confirm that messages have been received. When SMARt is compared with Conaware, there is a reduction of approximately $50\%$ in the resource search time.

Figure 5e shows the number of generated control messages, excluding the beacon. SMARt presented an overhead lower than the other approaches, and it required only a few messages to carry out the search. The overhead message is for the maintenance of the built-in overlay network. As a result, SMARt has achieved a reduction of $60\%$ in the number of messages compared to Conaware and a reduction of approximately $100\%$ when compared with the simplistic approach.

The solution proposal demonstrated a low search time (approximately 0.5 ms) to seek resources in one hop and approximately 0.9 ms to seek resources in more than one hop. Furthermore, SMART was able to provide high availability of resources (about $87\%$) with a low overhead (about 168 messages).

## V. CONCLUSION

In this paper, we proposed SMARt, a peer-to-peer protocol for search and management in the vehicular mobile Cloud, without the support of a roadside infrastructure. For achieving that, vehicles must be self-organized and collaborate with each other so that they can manage and share their resources. We compared SMARt with two other solutions, and showed that our proposed approach presented a low search time, approximately 0.5 ms, to seek resources in one hop and approximately 0.9 ms to seek resources in more than one hop. Furthermore, there is a high degree of availability of resources in SMARt - about $87\%$. In future work, we aim to adapt SMARt to operate in urban scenarios.

## REFERENCES


[1] T. Zhang, R. E. De Grande, and A. Boukerche, "Vehicular cloud: Stochastic analysis of computing resources in a road segment," in *Proc. of the 12th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks*, 2015, pp. 9–16.

[2] M. Whaiduzzaman, M. Sookhak, A. Gani, and R. Buyya, "A survey on vehicular cloud computing," *Journal of Network and Computer Applications*, vol. 40, pp. 325 – 344, 2014.

[3] M. Gerla, "Vehicular cloud computing," in *2012 The 11th IEEE Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2012, pp. 152–155.

[4] S. Bitam, A. Mellouk, and S. Zeadally, "Vanet-cloud: a generic cloud computing model for vehicular ad hoc networks," *IEEE Wireless Communications*, vol. 22, no. 1, pp. 96–102, February 2015.

[5] F. Ahmad, M. Kazim, A. Adnane, and A. Awad, "Vehicular cloud networks: Architecture, applications and security issues," in *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, Dec 2015, pp. 571–576.

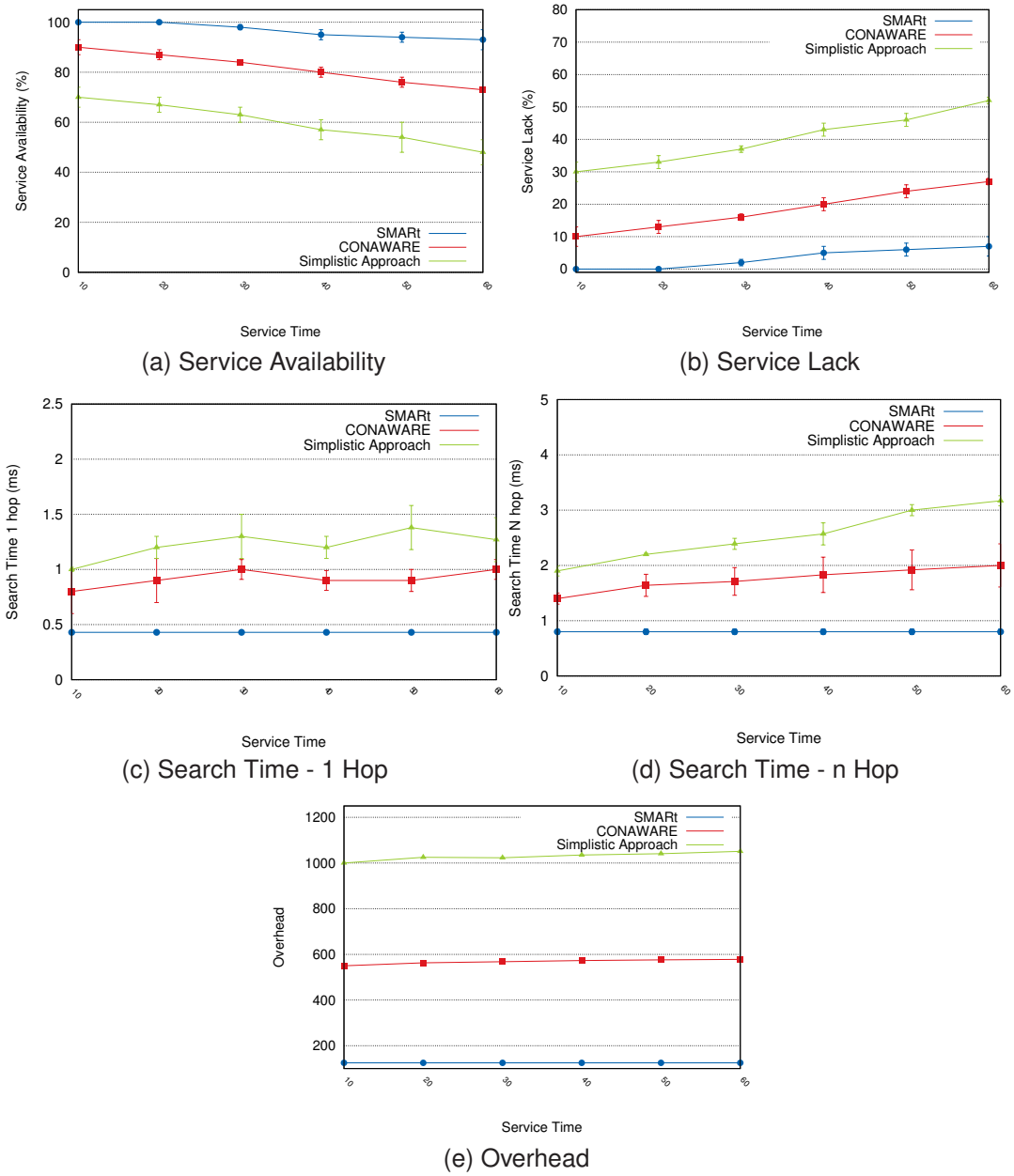


Fig. 5. Simulation Results of the Performance Analysis Conducted in a Realistic Highway Scenario.


[6] M. Eltoweissy, S. Olariu, and M. Younis, *Towards Autonomous Vehicular Clouds*. Springer Berlin Heidelberg, 2010, pp. 1–16.

[7] S. Olariu, T. Hristov, and G. Yan, *The Next Paradigm Shift: From Vehicular Networks to Vehicular Clouds*. John Wiley and Sons, Inc., 2013, pp. 645–700.

[8] D. Bernstein, N. Vidovic, and S. Modi, "A cloud paas for high scale, function, and velocity mobile applications - with reference application as the fully connected car," in *Proc. of the Fifth International Conference on Systems and Networks Communications*, 2010, pp. 117–123.

[9] J. Son, H. Eun, H. Oh, S. Kim, and R. Hussain, "Rethinking vehicular communications: Merging vanet with cloud computing," in *Proceedings of the 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)*, ser. CLOUDCOM '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 606–609. [Online]. Available: http://dx.doi.org/10.1109/CloudCom.2012.6427481

[10] S. Olariu, T. Hristov, and G. Yan, *The Next Paradigm Shift: From Vehicular Networks to Vehicular Clouds*. Wiley-IEEE Press, 2013, pp. 645–700.

[11] K. Mershad and H. Artail, "Finding a star in a vehicular cloud," *Intelligent Transportation Systems Magazine, IEEE*, vol. 5, no. 2, pp. 55–68, Summer 2013.

[12] R. E. Siba, T. Atchian, J. B. Abdo, R. Tawil, and J. Demerjian, "Connectivity-aware service provision in vehicular cloud," in *Proc. of the International Conference on Cloud Technologies and Applications (CloudTech)*, June 2015, pp. 1–5.

[13] R. E. Sibai, T. Atechian, J. B. Abdo, J. Demerjian, and R. Tawil, "A new software-based service provision approach for vehicular cloud," in *Proc. of the Global Summit on Computer Information Technology*, June 2015, pp. 1–6.

[14] SUMO, "Sumo - simulation of urban mobility," last visited in March, 2015, http://sumo.sourceforge.net/.

[15] C. Sommer, S. Joerer, and F. Dressler, "On the Applicability of Two-Ray Path Loss Models for Vehicular Network Simulation," in *Proc. of the IEEE Vehicular Networking Conference (VNC)*, 2012, pp. 64–69.