Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 2, February 2015, pg.176 – 187

RESEARCH ARTICLE

An Ontology-Based Approach for Optimal Resource Allocation in Vehicular Cloud Computing

Shahram Mazloom[1], Malihe Mohadesipour[2], Hamideh Babaei[3]

[1]M.Sc Islamic Azad University of Mahallat Branch, Iran

[2]M.Sc Islamic Azad University of Mahallat Branch, Iran

[3]Dr. Islamic Azad University- Naraq Branch, Iran

[1] emshahram@yahoo.com, [2] malihe_mohadesi@yahoo.com, [3] hamidehbabaei@yahoo.com

Abstract*— *In recent years, Cloud Computing has helped to optimal use of resources computing of car. Vehicular Cloud Computing is a new model of Cloud Computing that uses cloud computing to increase the capability of car. In this paper we use the technology, Vehicular Cloud Computing, to offer services with good quality of service to vehicles in moving. Services that are requested may be simple or compound. In simple services, we can measure the quality of service would deliver to the vehicle and then found proper service and offered it to the vehicle; but in combined services, the service tree must be constructed firstly and then used it for measuring the quality of service. Because it may take several providers, provide same services, selecting provider is important. To do this, we have been used the greedy and genetic algorithms and compared their products. In additional, we have included knowledge to Cloud using ontology until can offer the most suitable services to vehicles.

Keywords*— *Cloud Computing, Vehicular Network, Vehicular Cloud Computing, Resource Allocation, Compounded Service.

I. INTRODUCTION

Cloud Computing (CC) has become an important issue nowadays in the world, because it has ability to provide dynamic and flexible IT infrastructure, computing environments quality assurance and configurable software services. In fact, Cloud computing provides easy access to a computing resources collection, regardless provider location and delivery manner[1].

Vehicular Cloud Computing (VCC) is a technology that uses pay-per-use property of Cloud computing to help the vehicular network drivers. This technology offers several low-cost computing services for vehicle drivers to reduce traffic congestion, accidents, travel time and environmental pollution, also assures low energy consumption during use of the software, platform and infrastructure services[2].

Vehicular Cloud (VC) can offer its own resources by pure partnership and integrated management. Today, there are many vehicles on streets, highways and parking that have many unused resources. Currently, these resources are simply wasted, while can be used to provide public services. These features make the vehicles as good candidates for the formation of nodes in a Cloud computing network[3].

Resource allocation is a fundamental activity in Cloud computing environment. Since the Cloud computing, are provided services to users by virtual machines that affected by factors such as workload, time and crowding, like other servers[4]. When a server cannot provide a service according to client request, Service Level Agreement (SLA) is violated and response time is became longer[5]. Therefore, the purpose of dynamic virtual resource allocation is reduce response time, improve quality of service and fault tolerance[2].

In VCC, such as CC, resource allocation with optimal quality of service (QoS) is important and is motivated to increase quality of service and trying to safety drive on roads and streets.

QoS[6] attributes are usually part of SLAs. These agreements are contracted between service providers and clients in Cloud computing environments, and the server are obliged to provide service with QoS in the agreement[2,7]. In our proposed method, when a vehicle requests a service, this service may be simple or compound. Compound service, may be created by simple or other compound services. This structure is like a tree with multiple levels. Simple services are be selected according to QoS and other parameters such as distance and delay and then presented this service to the applicator vehicle. But about compound services, service tree must be formed firstly and then the final service is presented. Building a tree that it final QoS be equal to or greater than the requested QoS is very important. In this paper, according to movement parameters of vehicles and using optimization algorithms, we provide a method to optimally select service providers for building the tree. Therefore, using an ontology-based query from a catalog system[8], optimum resource is selected and proposed.

The results of this method, shows used algorithm to construct the service tree, has a significant impact on QoS of combined service. It should be noted that, necessarily the most appropriate service is not a service with the best QoS. Because, users must pay amount use each service. It is important find simple services that create the compound service. In addition, should be considered a reduction in QoS due to factors when service delivered to vehicle.

In this article, in the second part, we discuss previous work done in resource allocation. Then in the third section, we propose a method for optimal resource allocation and measuring QoS in VCC and discuss ontology-based catalog system and allocation algorithm. In the fourth section, we indicate simulation results and evaluate them. Finally, we indicate conclusions and future works.

II. PREVIOUS WORK

Given that the issue of resource allocation is one of the most important techniques, so we discuss it in two parts, CC and VCC.

A. Resource Allocation in CC

In Cloud Computing, allocating resources to demand Cloud applications over the Internet is called Resource Allocation (RA). Starvation is a challenge in resource allocation. Cloud service providers must provide and manage services. Resource Allocation Strategies (RAS), emphasis to integrate Cloud provider activities to allocate resource according to the type and amount source of user demand. Request time and priorities are also as the RAS entries. However, should not forgotten competition for resources, distribution of resources, providing additional and lower resources.

B. Resource Allocation in VCC

Given that Cloudlets are not able to provide required computing resources alone, to overcome this limitation, we can use from multiple Cloudlets as service provider. Furthermore, the quality of Cloud services changes constantly, because number of vehicles and the distance are variable. Therefore, in the resource allocation time must be compared agreement parameters with them.

In Vehicular Cloud Computing, Vehicles can be associated through roadside units (RSU) with roadside Cloud and Cloudlets as local resources also response to vehicles through roadside units. Therefore, higher quality of simple services in Cloud or Cloudlet is equal to higher quality of compound services.

III. PROBLEM DESCRIPTIONS

Often vehicles may be moving in VCC. So this is important how to communicate vehicles with Cloud infrastructure. Establish a wireless internet connection between vehicles and Cloud resources is difficult. For this reason, establish Cloudlet or roadside Cloud with fewer resources on roadside in order to create a high speed link. In this case, a vehicle requests services from central Cloud directly when it cannot connect to any Cloudlet. Cloudlets can also connect to central Cloud. On the other hand, requested services may be simple or compound. To build a compound service, may be use from simple or other compound services. There are various methods to select simple services providers that can build compound service with them. In this paper, we construct various versions

of a service tree using different permutation of service providers that they offer same simple services and then select the best service tree according to its final QoS. To do this, we have proposed and designed an ontology-based catalog system. Because, a resource has several parameters that affect to QoS, thus, we consider their approximate affect on quality of primitive services and then calculate final QoS or in other words, delivered QoS to vehicle and will give the best service to vehicles.

In this section, we present a method for allocating resources to the above-mentioned conditions and then, we present how to do it, rules, assumptions, metrics, assessment and results. To do this, we consider a V2I VANET that uses the IEEE802.11 in its MAC layer. In addition, we consider Cloudlets on the roadside with their RSUs that vehicles can connect to them. Services are simple or compound.

A. Proposed Method

According to Fig. 1, in the proposed method, vehicles are connected with wireless to the nearest RSU. The Request Package is contained request service name, request SLA (QoS) and geo position of vehicle.

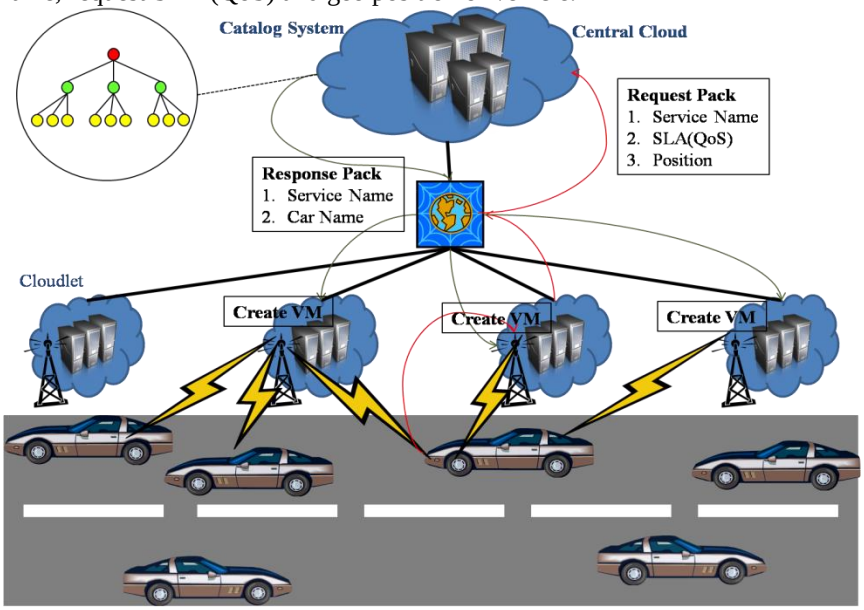

Fig.. 1 Steps of service request by vehicles an response to it

After the RSU received the Request Package, sends it to the central Cloud through Internet. In the central Cloud we have designed a catalog system. The catalog system has a knowledge base from offered services and their QoS and providers (Cloudlets), service tree of compound services and allocated services to vehicles. The catalog system, using its knowledge, first, checks the requested service to be compound or simple. If the requested service be compound then builds the service tree and according to simple service providers (Cloudlets), it selects the most suitable Cloudlets. After this, catalog system records specifications of Cloudlets and vehicles and sends a Response Package contained service name and requester vehicle name to the selected Cloudlets. The Response Package permits the Cloudlets to provide service. After the Cloudlets received the Response Package, create requested virtual machine and then start to service the vehicle.

Because vehicles are moving, changes their received QoS, for this reason, the catalog system must calculate quality of delivered service to vehicle. To do this, we consider parameters that affect to QoS, such as geographical distance and number of vehicles are client of each Cloudlet. Fig. 2 shows flowchart of the proposed method.

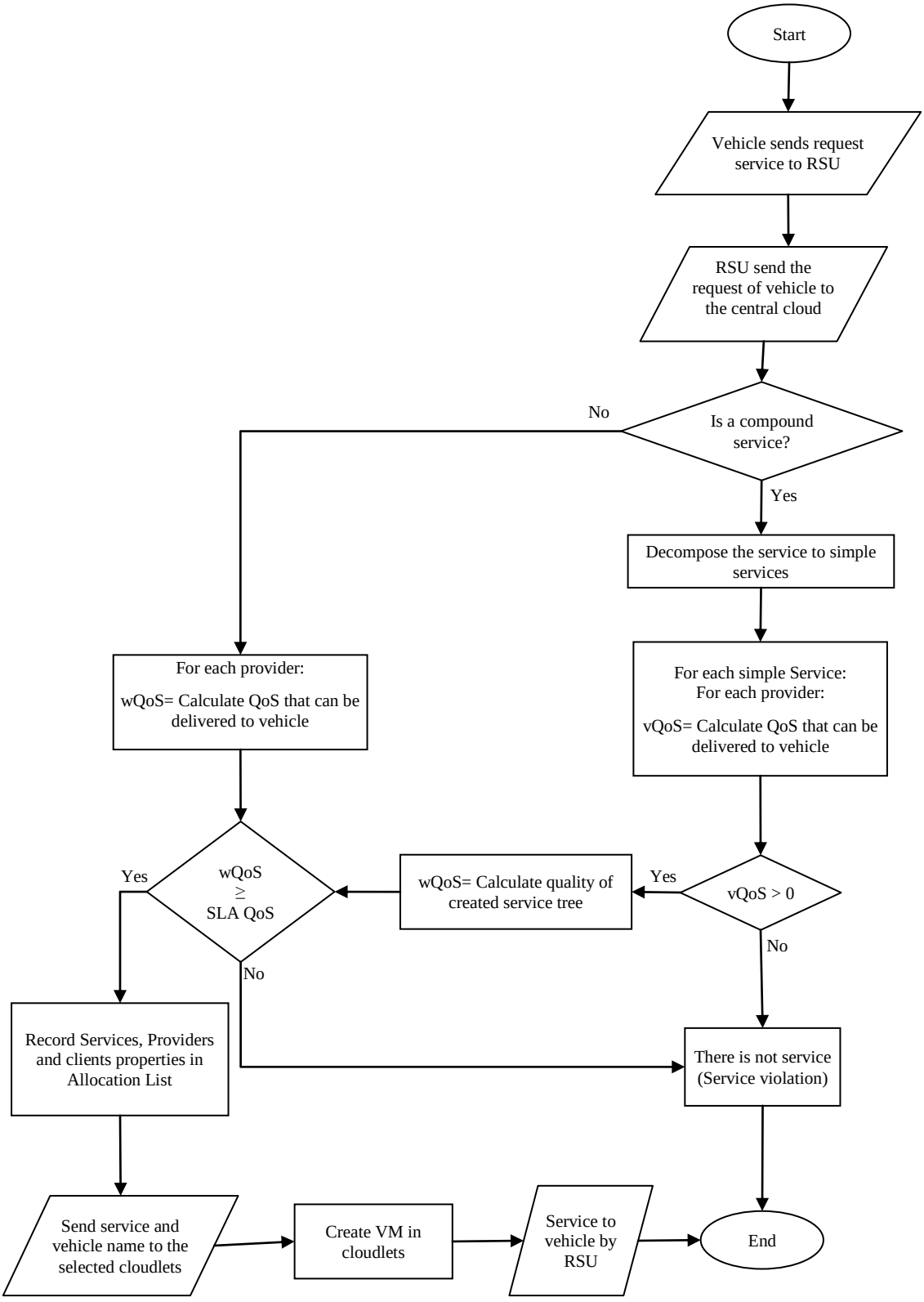

Fig. 2 Flowchat of the proposed method

1) Measuring Quality of Service:

Important parameters in this method are Quality of Service and geographical distance and number of clients for each Cloudlet. In this paper, we consider that QoS is Availability and two of the latter are impact parameters on QoS.

QoS: To measure the quality of simple and compound services, we have used from (Bardhan et al, 2012) but we have not considered redundant service and any service is a single point of failure.

Geographical distance: Distance is a space between RSU to vehicle. Distance is a factor that decreases QoS. To illustrate effect of distance on QoS, we have used the Formula (1):

Formula (1) $$\mathrm{QoS}_{\mathrm{Delivered}} = \frac{\mathrm{QoS}_{\mathrm{initial}}}{\mathrm{distance}_{\mathrm{GEO}}}$$

Number of clients: This parameter also decreases the QoS. To calculate effect of this parameter, first, the catalog system executes a query to achieve number of vehicles attached to RSU and then put it in the Formula (2). In this formula, k is number of clients.

Formula (2) $$\mathrm{QoS}_{\mathrm{Delivered}} = \frac{\mathrm{QoS}_{\mathrm{initial}}}{e^{k}} \ , \ k = 0, \dots, 200$$

By integrating the formulas (1) and (2), get Formula (3):

Formula (3) $$\mathrm{QoS}_{\mathrm{Delivered}} = \frac{\mathrm{QoS}_{\mathrm{initial}}}{\mathrm{distance}_{\mathrm{GEO}}.e^{k}} \ , \ k = 0, \dots, 200$$

2) Mechanism of Ontology Based Catalog System

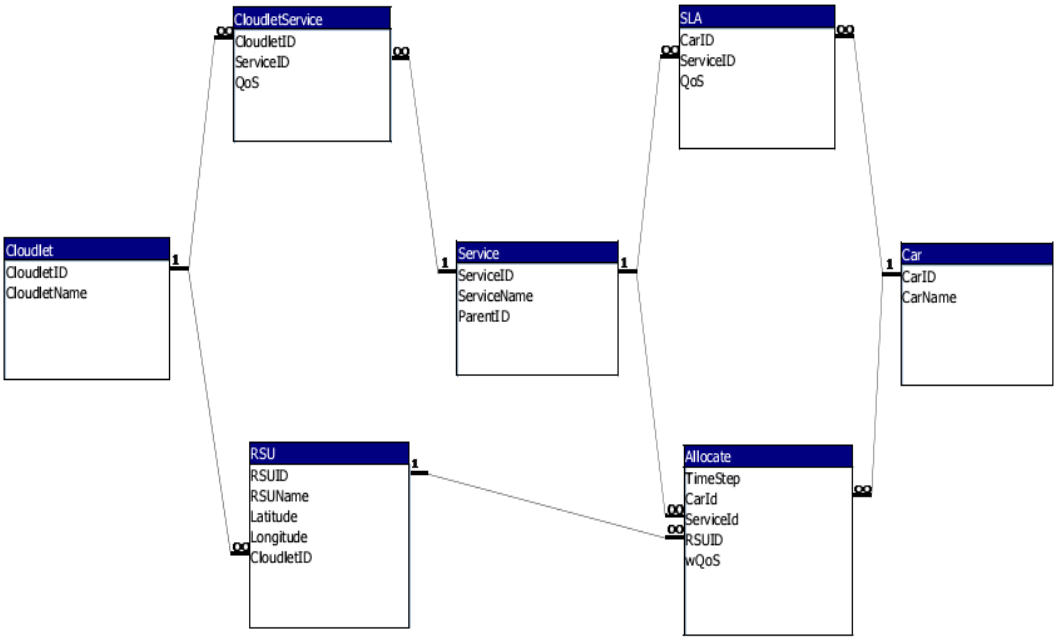

Fig. 3 The proposed catalog system

All items such as Cloudlets and their geographical position, offered services by each Cloudlet and their QoS, requests of vehicles with SLAs, which Cloudlet is servicing to which vehicles, are stored in catalog system with OWL.

As mentioned, compound services have a tree that its leaves are simple services (Fig. 4). Selecting Cloudlets as the simple service providers to replace instead leaves is important because in compound services, quality of final service must supply the SLA. For this reason, we use queries based on ontology in catalog system.

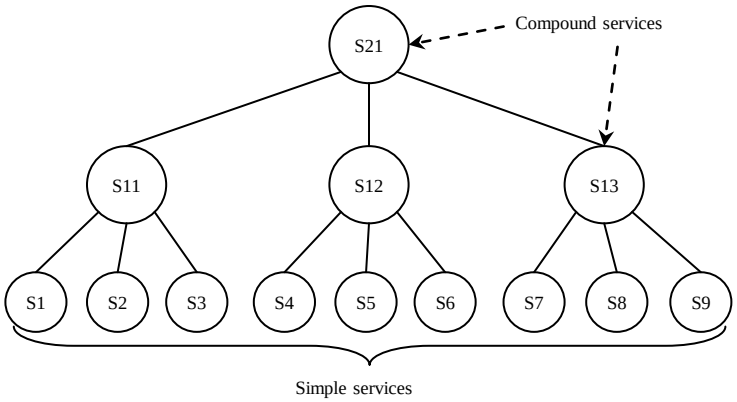

Fig. 4 A service tree

In this case, we use semantic model based on ontology that collects Cloudlets properties for allocating optimum service to vehicles. About simple services, may be exist services that their QoS supplies the requested QoS (QoS ≥ QoSRequested). But about compound services, we must calculate quality of its service tree. In these services, calculated QoS must be supplied the requested QoS (QoSCalculated ≥ QoSRequested). Moreover, we must consider effects of distance and number of clients. These amounts decrease the QoS. So, we must calculate quality of deliverable service (QoSDeliverable ≥ QoSRequested). The high level schema of our semantic proposed method based on ontology is shown in Fig. 5.

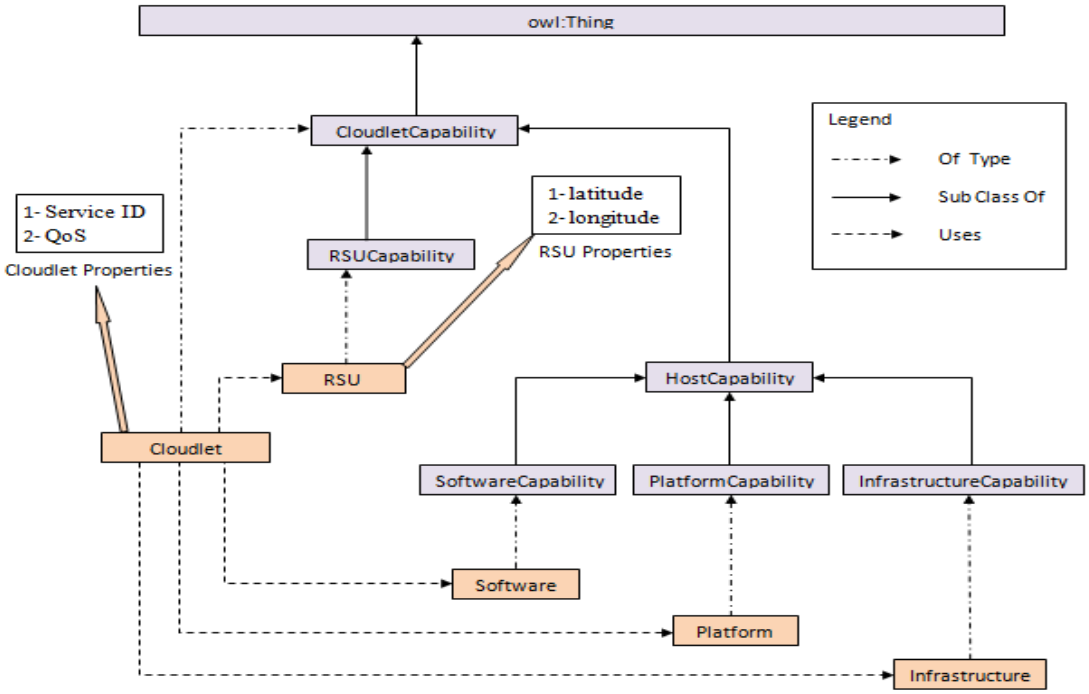

Fig. 5 Our method ontology with OWL

All defined classes are subclass of owl:Thing super class. CloudletCapability is subclass of owl:Thing; RSUCapability and HostCapability are subclasses of CloudletCapability and SoftwareCapability, PlatformCapability and InfrastructureCapability are subclasses of HostCapability. Any subclass inherits properties of its parent. RSU, Cloudlet, Software, Platform and Infrastructure

are instances of these classes. RSU is an instance of RSUCapability and has two properties, Latitude and Longitude that show geographical position of RSU. Cloudlet is instance of CloudletCapability and has two properties, ServiceID and QoS. The first characteristic is name of a service that Cloudlet provides and the latter indicates quality of the service.

3) Resource Allocation Algorithms

As mentioned, to allocate compound services we must create optimum service tree. For this purpose, we use Genetic and Greedy algorithms.

- Create Service Tree Using Genetic Algorithm: As regards, simple services create compound services, thus, we consider compound services as chromosome and each simple service as gene in this algorithm. Because more than a Cloudlet provides same services, we can create several chromosomes for each service. These chromosomes are different genetically. This method creates several same compound services with different QoS that we can select optimum compound service. There are several selection methods such as selecting maximum, minimum of maximums, random of maximums. In this paper, we use maximum and minimum of maximums as selection methods. Fig. 6 shows the Cloudlets as service providers or genes that create chromosomes. In this Fig., chromosomes sorted by deliverable QoS. In this example, we assume that requested QoS is 68%.

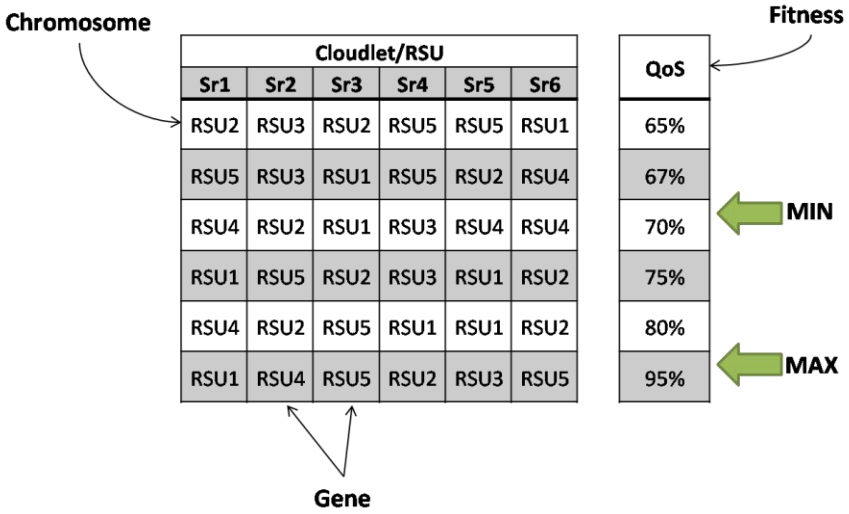

Fig. 6 Create service tree using Genetic algorithm

- Selecting Minimum of Maximum According to SLA (GA-Min): In the selection method, it is selected the first row of ascending sorted list that its QoS (Fitness) is greater than or equal to SLA. For example, in Fig. 6, if requested QoS be 68% then will be selected the third row (Chromosome).
- Selecting Maximum of Maximum According to SLA (GA-Max): In the selection method, it is selected the last row of ascending sorted list that its QoS (Fitness) is greater than or equal to SLA. For example, in Fig. 6, if requested QoS be 68% then will be selected the last row (Chromosome).
- Create Service Tree Using Greedy Algorithm: In this method, to create service tree, Cloudlets are selected with the highest QoS. The method offers a compound service with the highest QoS (Fig. 7). The offered QoS may be much greater than SLA that service consumer must pay its fees. But in speed and time, Greedy algorithm is much faster than Genetic algorithm. Greedy algorithm offers just one selection; while Genetic algorithm offers various selections. Fig. 7 shows an example that requested QoS is 68%.

Cloudlet/RSU						QoS
Sr1	Sr2	Sr3	Sr4	Sr5	Sr6	
RSU2	RSU3	RSU5	RSU1	RSU5	RSU1	97%

Fig. 7 Create service tree using Greedy algorithm

IV. SIMULATION AND PERFORMANCE EVALUATION

In this paper we use SUMO Simulator to implement vehicular network. We assumed vehicles speed is constant and equal. RSUs have been arranged so that they overlap.

A. Simulation Components

The used components in the simulation include roads, vehicles, Cloudlets, RSUs, central Cloud and services.

B. Simulation Scenarios

Road length is 100km; there are 50, 100, 150, and 200 vehicles; 10 Cloudlets and 10 RSU in roadside (Fig. 8).

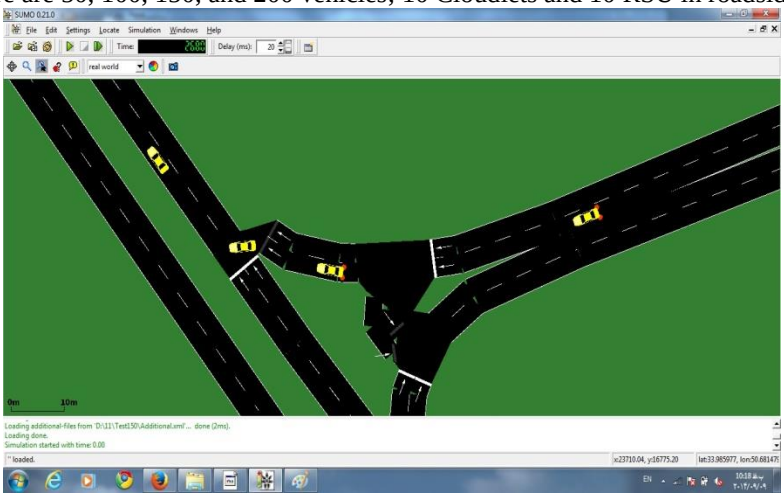

Fig. 8 Vehicle movement on road

C. Simulation Metrics

We considered following metrics to analyze simulator output file and evaluate methods:

- Average of Cloudlets efficiency: The Cloudlet workload is an important factor in determining QoS, thus we have calculated average Cloudlets workload in Formula (4). In the Formula, c is number of Cloudlet and n is number of vehicles.

  Formula (4) $$Cloudlets\ Workload\ Average = \frac{\sum_{i=1}^{c}(RSU\ workload)_i}{n}, c = 10$$

- Cloudlets service rate: We calculate average of servicing time for each service to vehicles and then calculate average them in Formula (5). In following Formula, t is the total of time measurement; s is the total of requested services and n is number of vehicles.

  Formula (5) $$Service\ Time\ Average = \frac{\sum_{i=1}^{n}\frac{\sum_{j=1}^{s_i}\frac{Count\left(t_{ij}\ \left[QoS_{ij}>0\right]\right)}{t_{ij}}}{s_i}}{n}$$

- Average of provided QoS: we calculated quality of any service to any vehicle and then average quality of all services to any vehicle and finally, measured quality of all services to all vehicles. All of them are shown in Formula (6).

  Formula (6) $$Service\ Time\ Average = \frac{\sum_{i=1}^{n}\frac{\sum_{j=1}^{s_i}\frac{\sum_{k=1}^{t}\mathrm{QoS_{ijk}}}{t_{ij}}}{s_i}}{n}$$

- Average of Service Violation: We need to measure the times that there were request services but there were no Cloudlets to give service them. Formula (7) indicates calculating method.

  Formula (7) $$Service\ Violation\ Average = \frac{\sum_{i=1}^{n}\frac{\sum_{j=1}^{s_i}\frac{Count\left(t_{ij}\ \left[QoS_{ij}=0\right]\right)}{t_{ij}}}{s_i}}{n}$$

- Cost of Vehicle: We calculate cost of each vehicle in all times that used services and then measure the average them. Formula (8) indicates all them.

Formula (8) $$Average\ of\ Service\ Time = \frac{\sum_{i=1}^{n} \frac{\sum_{j=1}^{s_i} \frac{\sum_{k=1}^{t} \text{Price}_{ijk}}{t_{ij}}}{s_i}}{n}$$

V. Simulation Results

A. *First Metric Results*

Chart1 shows that the average of Cloudlets service when Greedy algorithm is used is further than Genetic algorithm and GA-Max and GA-Min comparing indicates GA-Max is better than GA-Min.

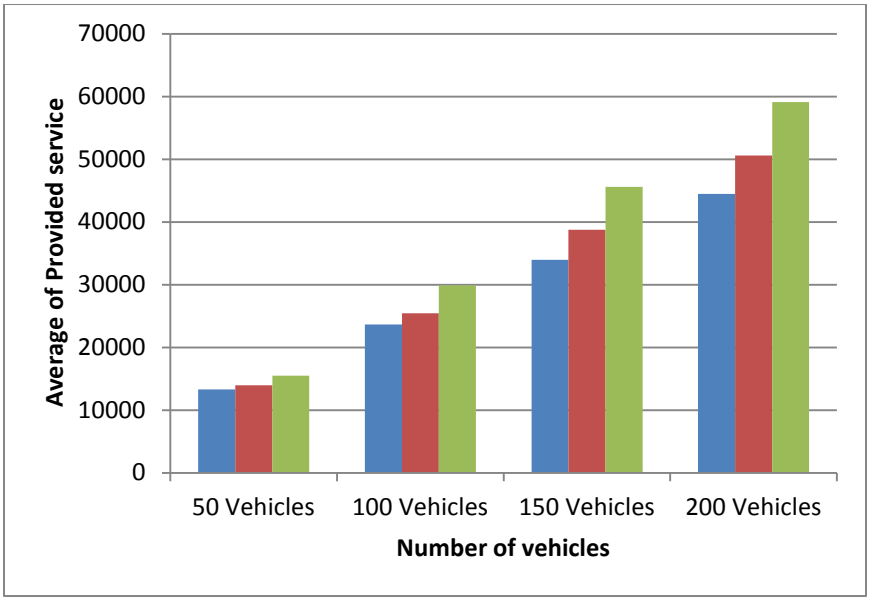

Chart 1: First metric results

B. *Second Metric Results*

Chart2 shows that the Cloudlets service rate when Greedy algorithm is used is better than Genetic algorithm and GA-Max and GA-Min comparing indicates GA-Max is better than GA-Min. In addition, increasing in number of vehicles causes to reduce GA-Min more than GA-Max and GA-Max more than Greedy.

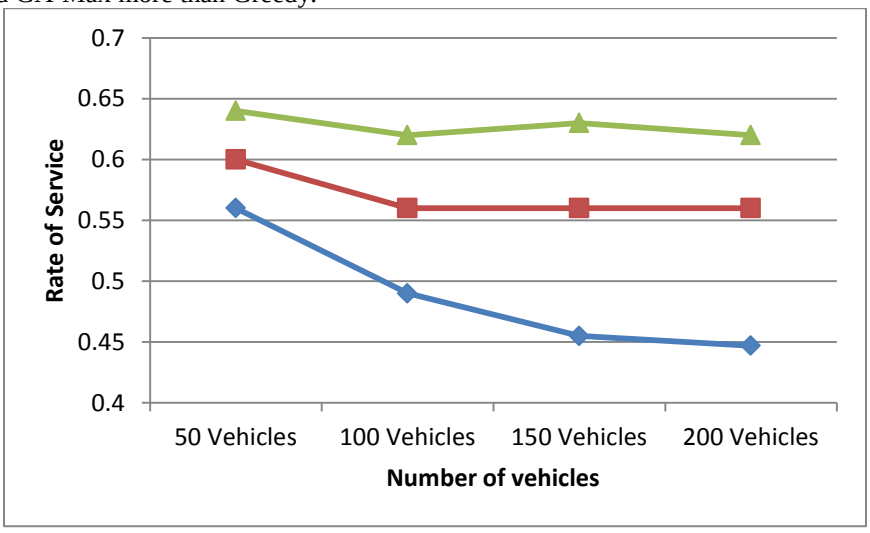

Chart 2: Second metric results

C. Third Metric Results

Chart 3 shows that average of QoS when Greedy algorithm is used is greater than GA-Max is greater than GA-Min. In addition, increasing in number of vehicles causes to reduce GA-Min more than GA-Max and GA-Max more than Greedy. As the Chart shows, Greedy is the nearest to SLA.

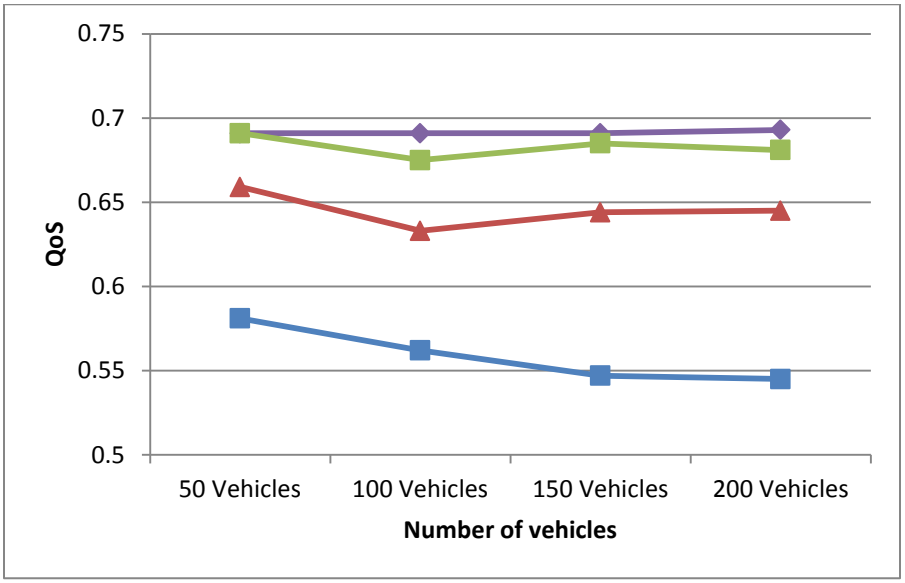

Chart 3: Third metric results

D. Fourth Metric Results

Chart4 shows that the service violation in Greedy is less than GA-Max and less than GA-Min. In addition, increasing in number of vehicles causes GA-Min to grow exponentially, while growth of the other states is almost constant.

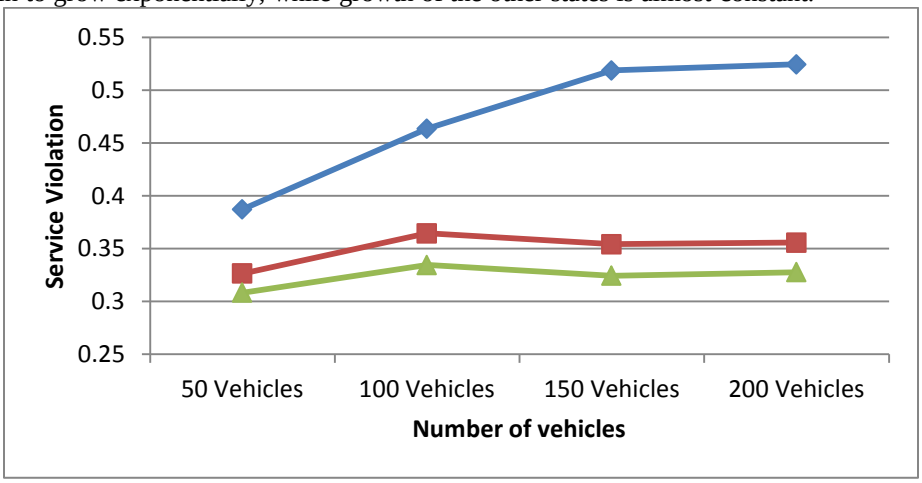

Chart 4: Fourth metric results

E. Fifth Metric Results

Chart5 shows that vehicles have the most payment in Greedy and amount decrease in GA-Max and again decrease in GA-Min. Of course, in different ways, by increasing the number of vehicles, their curves are closed together more.

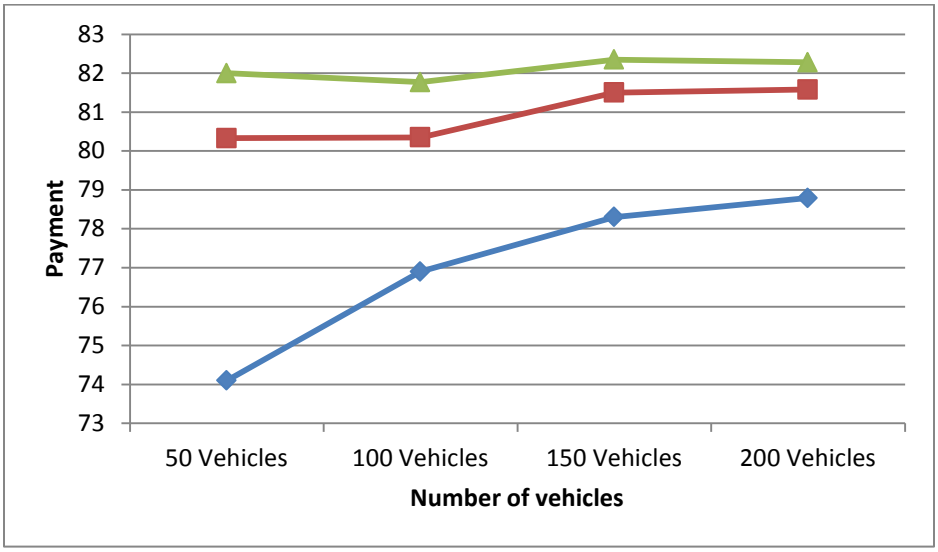

Chart 5: Fifth metric results

F. Comparison between Algorithms

Table 1 shows an overall assessment of the results.

Table 1: Results of the algorithms for all metrics

	Average of Cloudlets service	Rate of Cloudlets service time	Average of QoS	Average of service violation	Average of payment
Greedy	High	High	High	Low	High
GA-Max	Medium	Medium	Medium	Medium	Medium
GA-Min	Low	Low	Low	High	Low

VI. CONCLUSION AND FUTURE WORKS

We included ontology to VCC for making intelligent it. We created composite services according to requested QoS using ontology. It seems by providing appropriate services to vehicles, productivity of Cloudlets and their service rate will be increased and service violation and response time reduced; but the vehicles must pay more cost instead. For this purpose we used Greedy and Genetic algorithm. But it can be used other methods and algorithms such as learning automata and neural networks. In addition, we must consider service migration from a Cloudlet to another Cloudlet. To make composite services, we used services which all of them were single point of failure. In addition it can be used redundant services and indicated to its impact on QoS.

REFERENCES

[1] National Institute of Standards and Technology, The NIST definition of cloud computing, Recommendations of the National Institute of Standards and Technology, NIST special publication 800-145, 2009.

[2] Hewlett-Packard Enterprise Services, *A Mechanism to Measure Quality-of-Service in a Federated Cloud Environment*, published by Tulsa, vol. 918.625.1833, 2012.

[3] Olariu S, Hristov T and Yan G, *The Next Paradigm Shift: From Vehicular Networks to Vehicular Clouds*, Published by The Institute of Electrical and Electronics Engineers, 2013.

[4] IJACSA. *A Survey on Resource Allocation Strategies in Cloud Computing*. Published by Advanced Computer Science and Applications, vol. 3, No.6.PP: 97-104, 2012.

[5] *Queuing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science applications, published by Computer Network of Wiley*, pp: 894-904, IEEE, 2006..

[6] *QoS for Cloud Computing*, Coopration Territoriale. University de Pau, PP: 8-15, Pire Grid, 2011.

[7] International Computer Advanced (IJCA), *Mobile Agent as an Approach to Improve QoS in Vehicular Ad-hoc Network*, published by Special Issue on Mobile Ad-hoc Networks MANETs, PP:67-70, 2010.

[8] Bernstein D, et al, *A Cloud PAAS for High Scale, Function and Velocity Mobile Applications-with Reference Application as the Fully Connected Car*. 5th International Conference on Systems and Networks Communications (ICSNC), pp: 117-123, 2010.

