

Utilization of nested clustering in VANET to reduce data loss in mobile cloud computing

Erfan Arzhmand¹  · Hossein Rashid² · Mohammad Javad Fazel Ashrafi³

Received: 13 February 2016 / Accepted: 10 December 2016 / Published online: 28 December 2016
© The Author(s) 2016. This article is published with open access at Springerlink.com

Abstract Over the recent few years, most of network service providers have tried to expand their services all around by newly emerged technologies in network, especially mobile cloud computing. Most emerging cars and vehicles with powerful computers and storages, can be equipped with such new technologies such as vehicular ad hoc network radio, a new wireless technology. These capabilities lead network service providers to use these potentials to expand their network services such as Internet on the roads. It seems there are enough materials and peripherals to design a convenient network but every new architecture will have its limitations and problems, such as data loss, in performance. In this paper at first, we propose a design to find a proper cluster head in a group of vehicles. Then, based on some mathematical and physical methods and criteria, we demonstrate that nested clustering can reduce data loss in real streets and at equipped traffic-light crossroads, noticeably. Finally, we perform the simulation experiments to demonstrate the efficiency of our proposed approach in comparison with other approaches.

Keywords VANET · Nested clustering · Cloud computing · Mobile cloud computing

✉ Erfan Arzhmand
Erfanarj@gmail.com

Hossein Rashid
Hossein.Rashid.IR@ieee.org

Mohammad Javad Fazel Ashrafi
St_m.fazel@urmia.ac.ir

¹ Engineering and Technology Department, Shomal University, Amol, Mazandaran, Iran

² Computer Engineering Department, Islamic Azad University, Kerman, Iran

³ Computer Engineering Department, Urmia University, Urmia, Iran

1 Introduction

The ubiquitous requests to be connected to the networks, particularly internet and transferring large-volume data and huge files between internet and consumers, lead us to have more reliable and stronger connections and junctures [1]. Many devices such as smart phones need to be connected to the internet to be online for some applications. Moreover, new internet necessities and new cars and vehicles, equipped with computers and storages enjoying high market penetration rate lead internet service providers to expand their services even through streets. Theoretically, it seems that we can use these resources to design more reliable networks for vehicles on the streets. One of the most emerging technologies to integrate the mobile computer resources for expanding network resources is mobile cloud computing [2].

The mobile cloud computing (MCC) is defined by MCC Forum [3] as a reference to some infrastructures where data processing and storing happen outside the mobile devices and instruments. Based on this definition, Mobile Cloud applications move data storages and processing away from mobile phones to the cloud. This cloud contains not just smart phones resources but broader range of subscribers such as computers and so on.

According to the MCC structure and architecture, Dinh, HT, et al. proposed a categorized form for cloud services [4]. Based on this proposed form, we have three main layers for services in MCC. The first one is Infrastructure as a Service (IaaS). This layer contains hardware and network peripherals. The other layer is Platform as a Service (PaaS) which all operating systems are determined under this category and the last one is Software as a Service (SaaS) which software and applications to serve are enclosed in this field.

Based on above definition for MCC, we can use the other wireless technologies such as Vehicular Ad hoc Network (VANET) for mobile consumers to utilize vehicle resources like using their storages as IaaS [5]. In VANET topology, we have two main parts: the first, Nodes such as vehicles and the second one, Road Side Units (RSU). Nodes can connect together and are called Vehicle-to-Vehicle (V2V) or can connect to the exterior network such as Internet via RSU as a roadside infrastructure and are called Vehicle-to-Infrastructure (V2I) [6,7]. Due to connecting numerous nodes to the RSU, sometimes overhead on RSU occurs and it is not a sufficient state. Because of many solicitations they may collide and fail. To omit or reduce this problem, clustering topology for VANET has been proposed [8]. By this topology, some vehicles gather in a hypothetical cluster and just one member can make dialogue to RSU instead of all of cluster members. We have three kinds of members in a cluster. Cluster Head (CH) which can be the cluster manager, Cluster Gateway (CG) that can connect to other clusters as an interface and (CM) the rest of members, Cluster Member. The important challenge in this topology is to find the most adequate CH in a cluster [9].

The main purpose of clustering method is definitely the reduction of overhead. Therefore, a good clustering algorithm should focus on minimum number of clusters without increasing high communications over the network. It means that according to the clustering approach, we must generate the minimum of packages for V2I and V2V with reducing overhead on CH and RSU concurrently. Besides, we have dissemination at crossroads and intersections which is a significant problem because in cloud services, more stable connection has the highest priority to have the lowest data loss. Based on these problem assumptions, we propose Nested Clustering (NeCl) in our approach as an optimum solution to reduce overhead and data loss. According to NeCl, we can manage CH tasks and shift some of these tasks to Sub Cluster Head (SCH). A SCH is the Cluster Head of a Sub Cluster (SC). One of the most important conditions in a clustering method is to have minimum number of clusters. We definitely observe this condition because managing sub clusters are not on RSUs and their modifying and managing is done only on their cluster heads. It is reasonable to say that NeCl causes remarkable decreasing probability of overhead on CH because of reducing the number of tasks physically. Real streets have their criteria and conditions on intersections and we have some equipped traffic-light crossroads. We have considered these main facts in our proposal, too.

In this paper, we introduce our proposed approach, at first. Then we show how we select the most adequate CH. After that, we use some mathematical criteria and physical conditions to make Nested Clusters and we will show how this proposal works in a real environment theoretically. After that, we propose our simulation results by related diagrams

to show efficiency of our proposal in comparison with other approaches in this field. The last section concludes this paper and proposal.

2 Related works

There are very sparse approaches about cloud services on streets and their problems to transfer data from RSU to Vehicles [10]. In the following, we summarize three approaches which try to find suitable and optimum solutions. Then, we enumerate the main drawbacks of each one among all related works in this case.

One of these approaches has been proposed by Yu. Rong et al. [11]. It recommends a cooperative download/upload design with bandwidth sharing. In this approach, when applicant A is going to take a huge file from RSU, there are two steps. The first step is to find some vehicular neighbors such as B and C to set up a vehicular cloud for downloading cooperatively. In the next step, the file is split into some segments (three segments in this example) by RSU and each part is downloaded by vehicles A, B and C concurrently. At last, B and C send the separate file segments to A and A merges the file parts together finally. This proposal will increase the chance of getting a complete file remarkably. But to have an optimum approach there are some drawbacks. First, all of the vehicles should have negotiation with RSU. This is a shortcoming for this design because overhead will occur on RSUs for large amount of vehicles and their several solicitations. On the other hand, this proposal presumes that A, B and C will stay in a same way to complete the task and there is no dissemination among them. But in the real streets, dispersion may occur because of intersections, crossroads or traffic lights.

Arkian et al. [12], have proposed A Fuzzy Clustering-based Vehicular Cloud Architecture (FcVcA) to solve some inherent drawbacks such as overhead on RSUs. According to FcVcA, we should gather vehicles in some separated hypothetical clusters and then find the most proper CH for each cluster using fuzzy logic with some various factors like RSU link quality. By this method, we can see remarkable reduction of overhead on RSUs and because of most proper CH, based on simulation results we build a sufficient cluster to get data with lower data loss in comparison with other similar approaches. This method has some shortcomings, too. FcVcA is much suitable for highways and roads without intersections and any dissemination at crossroads and it has no idea to solve data loss on intersections. Another shortcoming is absolutely overhead on CH. Although we have well reducing overhead on RSUs, most of their tasks are shifted to CH. It is however, better than RSUs because the number of vehicles in a cluster is less than the number of vehicles included in working range of each RSU, but for

large amount of requests in the cluster, overhead may occur on CH.

The most similar approach to our proposal has been presented in [13]. In this approach, we use clustering method. Therefore, we select the most proper CH for each cluster using improved FcVcA, at first. Then, by using Genetic algorithm, the most adequate chromosome_x for data applicant X is generated, assisted by CH. After that, data is divided and some parts are moved to chromosome_x and other segments are transferred to X concurrently. The chromosome_x in this approach is a set of vehicles in a same cluster with X which the probability of their behavior at crossroad is almost similar to X. When dissemination occurs among cluster members at a crossroad, the rest segments of files which have not transferred yet will move from chromosome_x to X. By this method, based on its simulation results, data loss for huge files will be reduced at crossroads remarkably. Although this proposal has a sufficient solution for reducing data loss at crossroads, this method has been simulated on the Manhattan model without traffic lights on each intersections and it is a shortcoming for this proposal. Also, because of many processing and calculations, overhead on CH may occur and it includes another shortcoming for this proposal.

3 Proposed design

3.1 Overview

We design our system based on clustering in VANET. The first step to build a cluster is finding an adequate group of vehicles. In this step, we should choose a sufficient method depending on the circumstances and constraints of the problem. For instant, some of the proposed clustering methods are suitable for crowded paths [14, 15] or some other ones are convenient for a way with low density of vehicles [16]. On the next step, we should find a sufficient CH. CH is the manager of the cluster and controls the relations between the cluster and RSUs [17].

In this proposed approach, we want to reduce data loss that may occur even at an equipped traffic-light crossroad during file transferring from RSUs to vehicular applicants. For instance, when an applicant asks for a file from internet, RSU will reply. But if the asked file is huge, data loss may occur because of increasing the distance between RSU and applicant. Based on the proposed approach presented in [11], we can split requested file. According to [12], based on MCC, we send all parts of the requested file to the applicant and it seems a good idea to send its cluster concurrently to reduce the effect of increasing distance between RSU and applicant in transferring huge files with high efficiency. But at crossroads we have a different scenario. In this position, dissemination may occur between members of a cluster. To

reduce this, we must look for sufficient members within the cluster whose behavior at crossroad is similar to that of the intended applicants, as in [13].

According to our proposal, we use a pseudo-time division algorithm like the TDMA [18] to make a cluster on the street. When all vehicles between t_0 and t_1 go from a crossroad and enter a related street, RSUs classify them as a cluster and then choose one as an Initiator and send it the prepared list of cluster members and some information about the street like the number of the lanes. Initiator gets the position of each member and calculates the plurality of each lane. Then, Initiator chooses the most replete lane and by the average speed of cluster members, it selects CH among vehicles which are riding in the chosen lane. After that, Initiator takes both the cluster member list and the member position list to the selected CH.

Based on the quantity of the street lanes and following crossroad conditions, CH generates SCs and chooses one member of each SC as SCH. Then, it introduces SCHs to RSU and broadcasts them to own SC and gives the list of SC members to SCH. Also CH is the SCH of its lane. In this state, the cluster seems to be configured hierarchically. Therefore, for instance, when the member x_1 needs to get a file from RSU, it should ask its SCH at first. Then, SCH will ask this file from RSU and also leave own member list to it. RSU splits data in some equal predefined size and sends all parts to SC members with such algorithms as the Greedy Algorithm [19]. When the cluster arrives to the intersection, during unavoidable dissemination there is no data loss theoretically because cluster members which contain the file segments will follow the data applicant.

We consider some usual problems as follows in this proposed approach and have found out adequate solutions for them:

- Traffic lights at crossroad
- Changing the lane.

Although we care about the operating time by reducing the amount of calculations, reducing data loss on intersections during cloud computing and also bringing down the overhead on CH and RSUs are most important to our proposal. In a similar state, we prefer to satisfy these purposes against the operating time considerations.

3.2 Clustering and cluster head

B-1) Creating A Cluster

As mentioned in the overview section, clusters in this proposal are made by RSUs. In this case, RSUs cluster the entering vehicles to the street by specified time interval such as (t_0, t_1) . To recognize the entering vehicles, RSUs should calculate the coordinates of each vehicle. There are some

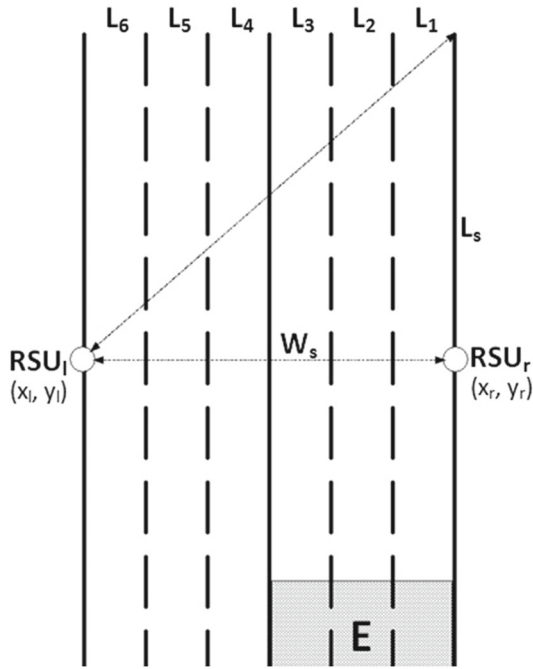


Fig. 1 The position of RSUs

famous methods to determine the coordinates of each vehicle like Received Signal Strength Indicator (RSSI), but based on some researches and experimental studies [20], RSSI is not reliable for multipath environments. Therefore, we decided to use the scheme of Ou, Chia-Ho [21] and its assumptions. It is GPS-free and most recent approach to calculate the coordinates of vehicles with high reliability and precision.

In this RSU-based localization scheme, we assume all vehicles equipped with VANET transceiver and digital odometer and compass. Based on Fig. 1, we assume that RSU_l and RSU_r with coordinates of (x_l, y_l) and (x_r, y_r), respectively, are installed on either side at the middle position. The coordinates of RSUs are determined by Global Positioning System (GPS). Also the radio range of RSUs (D) covers the width of the road. Therefore,

$$D \geq \sqrt{\left(\frac{L_s}{2}\right)^2 + W_s^2}. \tag{1}$$

In the meantime, the strip l_x (designated in Fig. 1 as l₁ to l₆) is the lane of the street. Also, if RSUs recognize the coordinates of a vehicle in district E, then add it to the list of entrances.

Based on Figs. 1 and 2, we can calculate the coordinates of vehicle Ve(x_{Ve}, y_{Ve}) as follows:

$$x_{Ve} = \frac{x_r^2 - x_l^2 + d_{lVe}^2 - d_{rVe}^2}{2x_r - 2x_l} \tag{2}$$

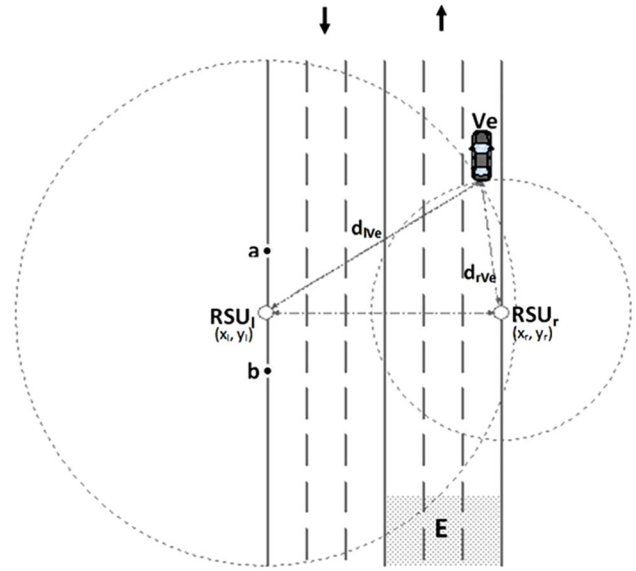


Fig. 2 Coordinating vehicle (Ve)

Also

$$y_{Ve} = y_l - \sqrt{-x_{Ve}^2 - x_l^2 + 2x_l x_{Ve} + d_{lVe}^2}, \tag{3}$$

where

$$(x - x_l)^2 + (y - y_l)^2 = d_{lVe}^2 \tag{4}$$

$$(x - x_r)^2 + (y - y_r)^2 = d_{rVe}^2. \tag{5}$$

As shown in Fig. 2, we have two hypothetical circles (4) and (5) which come from the beacon message and vehicle; Ve is exactly on their intersection with the following conditions:

$$y_l = y_r \tag{6}$$

$$x = x_{Ve} \tag{7}$$

$$y = y_{Ve}. \tag{8}$$

Although we can recognize the direction of vehicle Ve using the interior product to find the angle between the current movements vector **Ve** and the road direction **ab** in comparison with the angle between **Ve** and **ba**, we prefer to have a set of points stored in RSUs to recognize the direction of vehicle Ve because the interior product requires more calculations on RSUs. Also, we need more recognition to know which lane vehicle Ve is on. Therefore, it will be better to our proposal to use the sets of points instead of using some methods like the interior product.

RSUs continuously make clusters and get member for them in a specified time interval. When vehicle Ve arrives in the area E, RSUs check if this vehicle is free to join to a cluster; then they make it as a member of a cluster.

```

1: Coordinating()
2: {
3:   Coordinate  $Ve(x_{Ve}, y_{Ve})$ ;
4:   for ( $s=1$ ;  $s=n$ ;  $s++$ ) //maximum lanes of the street
5:   {
6:     if ( $Ve(x_{Ve}, y_{Ve}) \in lane_s$ )
7:     {
8:       return( $Ve(x_{Ve}, y_{Ve}), lane_s$ );
9:       break;
10:    }
11:  }
12: return 0;
13: }

```

Fig. 3 Pseudo-code of coordinating vehicle Ve

```

1: Clustering()
2: {
3:    $j=0$ ;
4:   for ( $t=t_0$ ;  $t=t_1$ ;  $t++$ ) //time interval
5:   if ( $Coordinating(Ve) \in E[]$ )
6:   if ( $Status(Ve)=FreeToJoin$ )
7:   {
8:      $ClusterMemberList[j][j] \leftarrow Ve, Coordinating(Ve)$ ;
9:      $j++$ ;
10:  }
11: }
12:  $i=rand() \% j+1$ ;
13:  $V_{init}=Ve_i$ ;
14: send ( $Ve_{init}, ClusterMemberList$ );
15: send ( $Ve_{init}, NumberOfLanes$ );
16: return 0;
17: }

```

Fig. 4 Pseudo-code of clustering

In Fig. 3, line 6 tries to find the lane of $Ve(x_{Ve}, y_{Ve})$. It must be compared with the set of points for each lane. These sets for each street are calculated and stored in RSUs previously.

As Fig. 4 shows, every vehicle being in a time interval and included in area E, and not being a member of other cluster, must join the new created cluster. After getting members at t_1 , RSUs select one of the members as initiator (lines 12 and 13). At last, RSUs send the list of cluster members and the number of the street lanes to Ve_{init} .

B-2) Selecting The Cluster Head

Ve_{init} , based on the information of members, should select the most adequate CH for this hypothetical cluster. It has the coordinates and exact lane of each vehicle as the basic information, but it needs the average speed of each vehicle to select CH precisely. In fact, it needs to know the value of adjacency of each vehicle to the average speed of cluster members.

To measure this adjacency, we ought to use Eq. (9) as follows:

$$AdS_{Ve} = \frac{(Av_{Ve} - Av_c)}{\sigma_c}, \tag{9}$$

```

1: SelectingCH()
2: {
3:    $plurality(lane_{CH})=0$ ;
4:   for ( $i=1$ ;  $i=n$ ;  $i++$ ) //n lanes
5:   if ( $plurality(lane_{CH}) < plurality(lane_i)$ )
6:    $lane_{CH}=lane_i$ ;
7:   elseif ( $plurality(lane_{CH}) = plurality(lane_i)$ )
8:    $Lane_{CH}=rand(lane_{CH}, lane_i)$ ; //choose one of two
9: }
10:  $CH= Lane_{CH}.member_i$ ;
11: for ( $j=2$ ;  $j=m$ ;  $j++$ ) //m members in lane_{CH}
12: if ( $AdS(Lane_{CH}.member_j) < AdS(Lane_{CH}.member_i)$ )
13:    $CH= Lane_{CH}.member_j$ ;
14: elseif ( $AdS(Lane_{CH}.member_j) = AdS(Lane_{CH}.member_i)$ )
15:    $CH= rand(CH, Lane_{CH}.member_j)$ ; //choose one of two
16: }
17: return 0;
18: }

```

Fig. 5 Pseudo-code of selecting CH

where

$$Av_{Ve} = \frac{1}{3} \sum_{i=1}^3 v_i \tag{10}$$

$$Av_c = \frac{1}{n} \sum_{j=1}^n v_j \tag{11}$$

$$\sigma_c = \sqrt{\frac{1}{n} \sum_{j=1}^n (Av_{Ve} - v_j)^2}. \tag{12}$$

As Eq. (10) shows, we need only three samples of velocity for each vehicle to calculate the average speed, without more calculating and computing such as acceleration because we assume that we are in the urban area which contains vehicles with totally constant velocity. Av_c is the average speed of cluster in (11) and at last, σ_c is the standard deviation of average speed of Ve among n member of cluster. In (12) we omit Bessel’s correction because we have included whole members of the cluster [22].

To select the adequate CH, Ve_{init} calculates the number of members on each lane of the street (plurality), at first. Then, using the values of plurality and adjacency to average speed (AdS), it selects the most adequate CH of this cluster (Fig. 5).

Lines 8 and 15 in Fig. 5 indicate that if we stay on a point to prefer one of two equal variables, then we choose randomly.

When CH is selected, Ve_{init} sends the list of cluster members and their information, at first. Then, it broadcasts CH to all members of cluster and introduces it to RSUs.

3.3 Nested clustering

The next phase of our proposed approach is NeCl. Cluster Head (CH), based on the coordinates of cluster members and Next Crossroad Conditions (NCC), builds nested clusters as

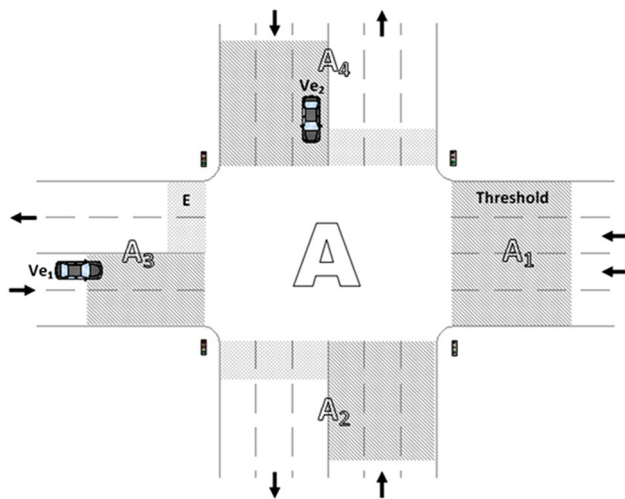


Fig. 6 Crossroad A

the Sub Cluster (SC). Then, it selects one of SC members as the Sub Cluster Head (SCH). It is clear that CH is one of SCHs in this procedure.

As Fig. 6 shows, we have crossroad A for instance. Crossroad A has four branches: A₁, A₂, A₃ and A₄. Each branch of crossroad A has its conditions. For instance, A₃ has two ways and each way has two lanes or A₁ has one way and four lanes. Vehicle Ve₁ is coming to crossroad A from A₃ and it has just two choices: turning right on intersection A to A₂ or turning left to A₄. Also, vehicle Ve₂ has two choices: turning right to A₃ and going straight to A₂. In the meantime, there is no choice to turn to A₁ at crossroad A.

At some urban crossroads, we have traffic lights and, therefore, this is another condition for a crossroad that we should consider. In addition, as we can see in Fig. 6, each branch has a threshold near the crossroad. In threshold field, vehicles cannot change their lanes. This fact is salutary during NeCl calculations.

Another condition for a crossroad is the weight of each branch. Based on previous gathered information, we know how many vehicles at a crossroad have been detected by RSUs in a time arrival. In Fig. 6, if RSUs detects x_1 , x_2 , x_3 and x_4 vehicles corresponding on branches A₁, A₂, A₃ and A₄, then we can calculate the weight of A_a as follows:

$$We(A_a) = 100 \left(\frac{x_a}{x_A} \right), a = 1, 2, 3, 4, \quad (13)$$

where

$$x_A = \sum_{a=1}^4 x_a. \quad (14)$$

```

1: CreatingSC()
2: {
3:   for (j=1;j=street.lane; j++){
4:     if (purpose(lanej,member)=purpose(lane(j-1),member))
5:       SC(j-1)[]←lanej.member;
6:     else{
7:       SCj[]←lanej.member;
8:       counter++;}
9:   for (i=1;i=counter; i++){
10:    if (CH∉SCi[])
11:      r=rand() % SCi[][.quantity+1
12:      SCHi=SCi.member;
13:    elseif (CH∈SCi[])
14:      SCHi=CH;
15:    }
16:  return 0;
17: }
```

Fig. 7 Pseudo-code of creating SC

The condition of weight and its proportion for each branch at a crossroad is sufficient for our proposal to distinguish and prefer more probable one among some similar choices.

Cluster Head (CH), based on NCC and the number of lanes, creates SCs. As Fig. 7 shows, CH creates SC based on the number of the lanes, at first. Also, the function Purpose() checks the destination of vehicles on lane_j based on NCC. If destinations are the same, then both member lanes will be in a same SC. The function Purpose() also recognizes the vague purposes as unequal state.

If vehicle Ve changes its lane after clustering, it should tell its related SCH to remove it from the list of SC and tell CH. Then, CH asks the new coordinates of Ve from RSUs and RSUs reply CH using Coordinating (). After that, based on the new coordinate, CH introduces it to the related SC.

For instance, if Vehicle Ve needs a file, it asks from the related SCH. SCH asks this file from RSUs and gives the list of its SC to RSUs simultaneously. RSUs split the file in presumed size if needed. After that, they send the parts of requested file to members of SC containing the name of applicant Ve. Then, SC members transfer those segments to Ve.

When CH touches the Threshold Area, as depicted on Fig. 8, it asks the state of traffic light from RSUs (C_{sl} , t_{sl}). C_{sl} is the color of traffic light and t_{sl} is the remaining time of being C_{sl} . As Pseudo-code of Fig. 8 shows, if CH arrives in green light, then it broadcasts to SCHs.

According to Fig. 8, t_{CH} is the required time for CH to arrive crossroad and it is calculated follows:

$$t_{CH} = \frac{y_{th}}{Av_{CH}}. \quad (15)$$

As Fig. 9 depicts, we can see the SCs of Cluster A. Also during t_{sl} , if each SCH recognizes that dissemination in SC at crossroad will occur, then it creates Sub SC (SSC) and defines a member of SSC as SSC Head (SSCH). If applicant

```

1: TrafficLight()
2: {
3:   Coordinate ( $Ve_{CH}$ );
4:   if ( $Ve_{CH}(x_{CH}, y_{CH}) \in ThresholdArea[]$ )
5:     AskTrafficLight( $C_{st}, t_{sl}$ );
6:   if ( $C_{sl} = Green$ )
7:     if ( $t_{CH} < t_{sl}$ )
8:       Broadcast( $t_{sl}$ );
9:   return 0;
10: }
    
```

Fig. 8 Pseudo-code of traffic light

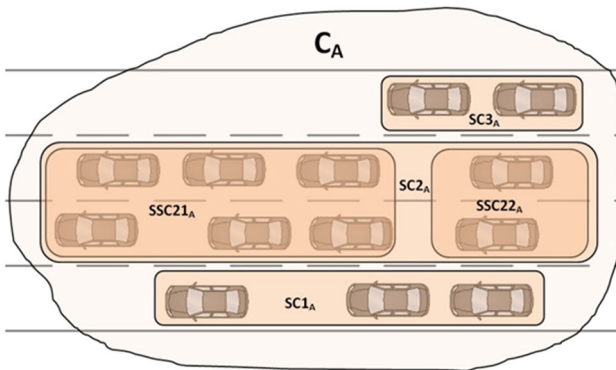


Fig. 9 Cluster and sub clusters

```

1: CreatingSSC()
2: {
3:    $SSC_i[] = 0$ ;
4:   for ( $i=1; i=SC.quantity; m++$ )
5:     if ( $t_i \geq t_{sl}$ )
6:        $SSC_i[] += member_i$ ;
7:   }
8:   if ( $SSC_i[] \neq 0$ )
9:      $SSC_2[] = SC[] - SSC_1[]$ ;
10:   if ( $SCH \in SSC_i[]$ )
11:      $SSCH_1 = SCH$ ;
12:      $r = rand() \% SSC_2.quantity + 1$ ;
13:      $SSCH_2 = SSC_2[.member_r]$ ;
14:   else
15:      $SSCH_2 = SCH$ ;
16:      $r = rand() \% SSC_1.quantity + 1$ ;
17:      $SSCH_1 = SSC_1[.member_r]$ ;
18:   Broadcast ( $SSCH_1 \& SSCH_2$ );
19: }
20: return 0;
21: }
    
```

Fig. 10 Pseudo-code of creating SSC

Ve is in SSC_1 , then all parts of data which SSC_2 members are carrying should be transferred to SSC_1 members. Figure 10 presents the Pseudo-code of creating SSCs and selecting SSCHs, respectively. Figure 10 also shows the method of selecting SSCH.

By NeCl, transferring big files to its applicant is done by reducing data loss at crossroads because NeCl predicts the causes of data loss and gives some solutions to have more reliable data transferring.

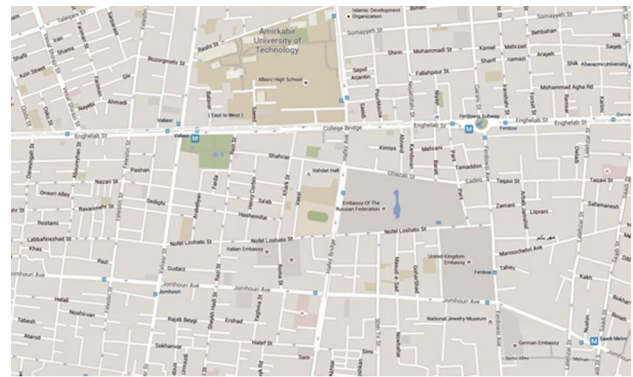


Fig. 11 Jomhour and Enqelab avenue

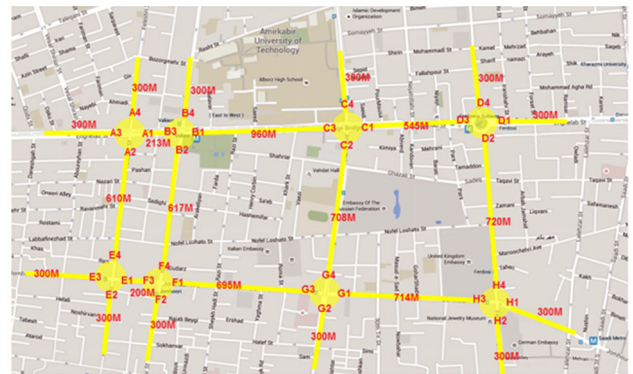


Fig. 12 Simulation area

4 Performances and simulations

To simulate our proposed approach, we have considered the reality. Therefore, we simulated our proposal in such real plan with the real scales. As depicted in Fig. 11, the simulation area is in zone eleven of Tehran city which contains a part of Jomhour and Enqelab Avenue and also some other related streets and roads [23]. Both of Jomhour and Enqelab Avenue are the famous roads in the middle of Tehran.

For simulating our proposed approach, we use both OMNet++ [24] as a network simulator and SUMO [25] as a traffic simulator. These tools connect and work together using VEINS [26].

As Fig. 12 shows, our simulation area contains 8 crossroads and their 22 related roads and also the length of each one. There are more intersections and crossroads but we use main roads. Also, our proposal needs crossroads to be equipped with traffic lights for possible evaluation. The table of branches of each crossroads and related lanes has been sorted in the Table 1.

Table 1 shows the main part of Crossroad Conditions for the simulation area. For instance, at crossroad A, there are two lanes for going out from A via A_1 and two lanes for coming into A via A_1 and also, at crossroad G, there are

Table 1 Conditions of the crossroads branches

Branch	Out	In	Branch	Out	In
A ₁	2	2	E ₁	3	0
A ₂	1	0	E ₂	1	0
A ₃	2	2	E ₃	0	3
A ₄	0	1	E ₄	0	1
B ₁	2	2	F ₁	3	0
B ₂	2	2	F ₂	2	2
B ₃	2	2	F ₃	0	3
B ₄	2	2	F ₄	2	2
C ₁	2	2	G ₁	3	0
C ₂	3	0	G ₂	3	0
C ₃	2	2	G ₃	0	3
C ₄	0	3	G ₄	0	3
D ₁	2	2	H ₁	3	0
D ₂	0	3	H ₂	0	3
D ₃	2	2	H ₃	0	3
D ₄	3	0	H ₄	3	0

three lanes for going out from G via G₂ and there is no lane for coming into G via G₂.

According to Table 1 and this simulation area, we have 22 roads and 68 lanes of which 18 lanes are for the input of the vehicles in this simulation scenario. Also it contains 8 equipped traffic-light crossroads. We presume all vehicles are equipped with transmitter radio that supports IEEE 802.11p and IEEE 1609 [27] standards. In addition, we assume Poisson distribution in traffic ratio [28]. The total time of simulation is 300s and we repeat each level of simulation for 1000 times and increase the certainty of the results using Monte Carlo method [29]. The assumptions and parameters of our simulation are sorted in Table 2.

Our simulation is focused on the content of the transferring file. We choose three contents for the mentioned file as 2MB, 4MB and 8MB to compute the probability of getting whole of file successfully. Also, we consider both factors of the total number of vehicles and the average speed of vehicles in the roads. Then, we compare our results with the similar results of FcVcA and Genetic method.

To calculate the probability of getting file (P_{get}), we use equation (16) as follows:

$$P_{\text{get}}(f_{\text{suc}}|f_{\text{sent}}) = \frac{P(f_{\text{suc}} \cap f_{\text{sent}})}{P(f_{\text{sent}})} \quad (16)$$

Since f_{suc} and f_{sent} are not the independent events, we use equation (16) which f_{suc} is the set of taken files successfully and f_{sent} is the set of sent files totally [30].

For the first level of simulation, presuming the velocity of all vehicles 40km/h, we consider the number of vehicles in the roads. Theoretically, when the quantity of vehicles

Table 2 Simulation parameters and their values

Parameter	Value
Simulation time	300 s
Road length	200–960 m
Number of vehicles	550–1350
Number of roads	22 (68 lanes)
Number of crossroads	8
Number of Traffic lights	22
Red light	15–45 s
Green light	15–45 s
Yellow light	3s
Speed of vehicles	10–45 km/h
Transmission ratio	5 Mbps
Size of message	100 Bytes
Safe distance between vehicles	2 m
Vehicle length	3 m
D_{th}	150 m

increases as a resource, the total successful transferred files will increase [31]. Our results depicted in Fig. 13 show this fact. When the total vehicles increase, the probability of getting file is increased but, based on our results, we can see some differences between the results of our proposed approach and the others. The main reason is absolutely vulnerability of FcVcA and Genetic method at equipped traffic-light crossroads. Although Genetic method has a solution at crossroads to reduce data loss, we should consider that Genetic method has not any solution for traffic lights and, therefore, dissemination may occur in clusters because of traffic lights. On the other side, FcVcA has not any solution even at crossroads and fragmented clusters. Therefore, it is reasonable if we see data loss in this proposal.

Our explanations for Fig. 13 can explain Figs. 14 and 15. But with a closer look, we can see if the size of packet increases, the diagram of our proposed approach and genetic method will take some more distance with FcVcA diagram. Although the Genetic method has vulnerability at equipped traffic-light crossroads, FcVcA has not seen crossroads in its proposed approach and this is the main reason for this meaningful distance between FcVcA diagram and the others when the size of file increases.

The next level of our simulation is focused on the velocity of vehicles. It is reasonable if the total velocity of vehicles increases, then data loss in vehicular cloud computing would increase. It is sufficient for us that we reduce data loss concurrently when the total speeds of vehicles increases. Figures 16, 17 and 18 declare our simulation results about the probability of getting file successfully when the total velocity of vehicles increases. As we can see in these diagrams, when the total velocity increases, data loss increases on all

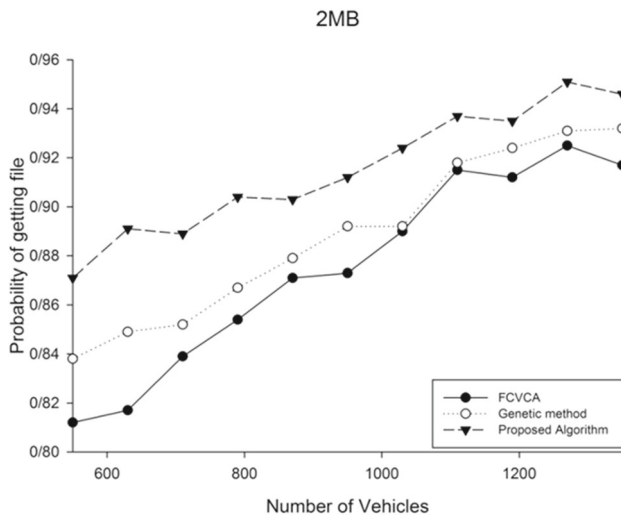


Fig. 13 Probability of getting file vs. number of vehicles (2MB)

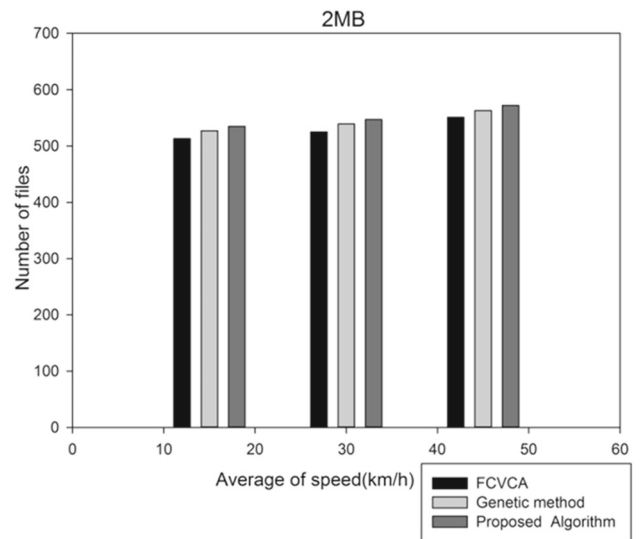


Fig. 16 Number of files vs. average of speed (2MB)

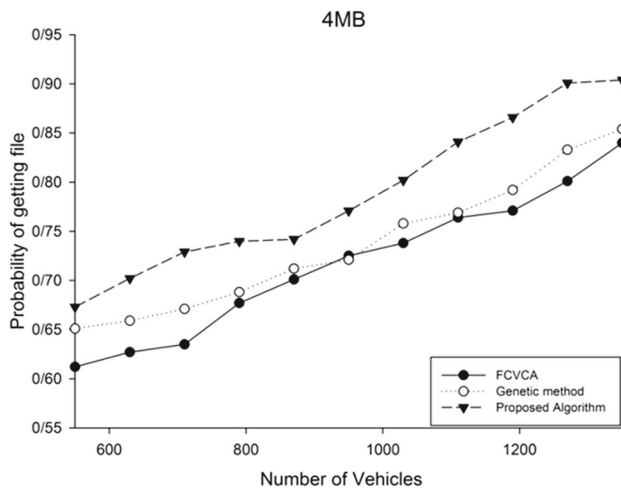


Fig. 14 Probability of getting file vs. number of vehicles (4MB)

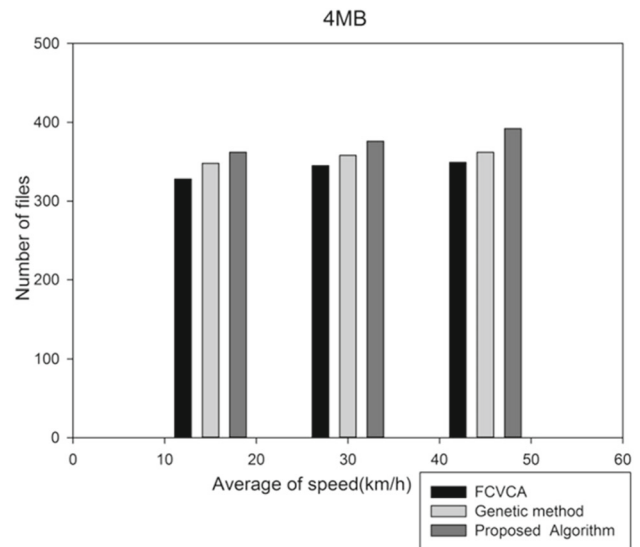


Fig. 17 Number of files vs. average of speed (4MB)

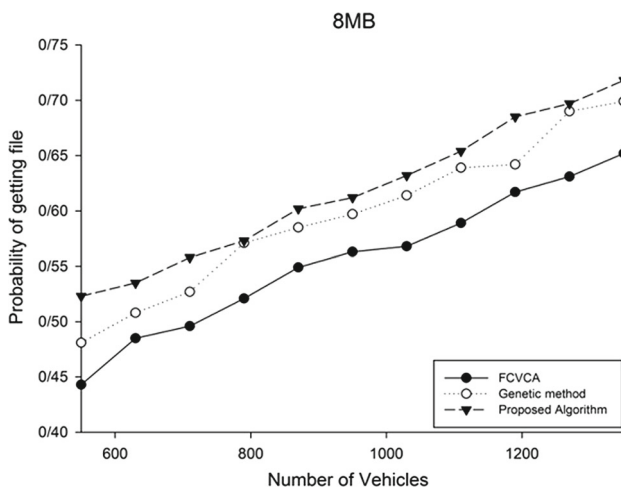


Fig. 15 Probability of getting file vs. number of vehicles (8MB)

compared methods. Although data loss is increased totally, the probability of data loss in our proposed approach is less than others remarkably. When a file is divided, each part is smaller. Therefore, the chance of getting it is increased. On the other hand, gathering all split parts of a file among some vehicles and merging them is difficult and it causes data loss to increase even when a part of a file is missed. All compared methods support this solution. But at crossroads the probability of data loss is increased because we have disseminations among vehicles. Hence, our proposed approach increases the probability of getting a file successfully because it can reduce data loss in comparison with other similar compared methods.

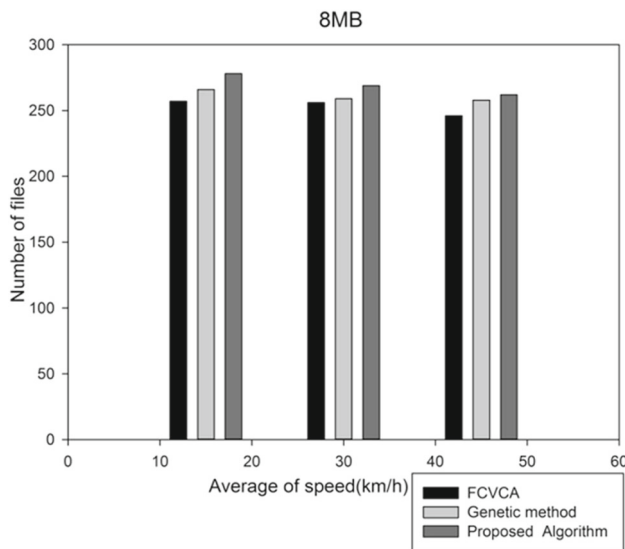


Fig. 18 Number of files vs. average of speed (8MB)

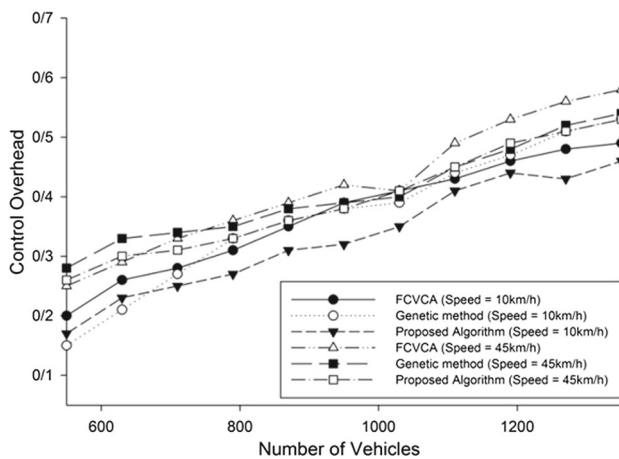


Fig. 19 Control overhead

Based on our proposed approach, having the adequate overhead is questionable in comparison with FcVcA and Genetic Method because we have many calculations and V2V and V2I negotiations to find sufficient CHs and cluster vehicles adequately.

To calculate and compare this factor, we should define the Control Overhead. Control Overhead is defined as the ratio of the total number of control messages to the total number of packets to make a cluster.

Figure 19 shows the comparison of this ratio among our proposed approach, FcVcA and Genetic method. As we can see, even though all three diagrams are really close together, the control overhead of NeCl has fewer ratios. Based on the control overhead for 10 and 45 km/h, although our proposed approach has many calculations and V2I and V2V dialog to perform the clusters, sub clusters and CHs, many calculations of Fuzzy logic increase the Control overhead of FcVcA

and Genetic method. This is one of our main reasons to use the other method to calculate instead of Fuzzy logic in our proposed approach.

5 Conclusion

At first, we defined MCC and VANET and also, detailed them and their relations. After that, we introduced and described some related works in this field. Then, based on clustering method in VANET, we introduced NeCl in VANET. Furthermore, using NeCl, we presented our proposed approach to reduce MCC data loss at crossroads, especially the equipped traffic-light crossroads using nested clusters, their adequate CHs and the crossroad conditions based on NeCl. Finally, we depicted and detailed the diagrams and schemes of our simulation results in comparison with two other similar methods.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Gantz, J., Reinsel, D.: The digital universe in 2020: big data, bigger digital shadows, and biggest growth in the Far East. IDC iView: IDC Anal. Futur. **2012**, 1–16 (2007)
- Whaiduzzaman, Md, et al.: A survey on vehicular cloud computing. J. Netw. Comput. Appl. **40**, 325–344 (2014)
- MCC-forum. Discover the world of Mobile Cloud Computing. London: mobile cloud computing forum (2011)
- Dinh Hoang, T., et al.: A survey of mobile cloud computing: architecture, applications, and approaches. Wirel. Commun. Mob Comput. **13**, 1587–1611 (2013)
- Mershad, K., Artail, H.: Finding a STAR in a vehicular cloud. Intell. Transp. Syst. Mag. IEEE **5**(2), 55–68 (2013)
- Bordley, L., Cherry, CR., Stephens, D., Zimmer, R., Petrolino, J.: Commercial motor vehicle wireless roadside inspection pilot test, Part B: Stakeholder perceptions. In: 91st annual meeting of the transportation research board (2012)
- Yang, X., Liu, L., Vaidya, NH., Zhao, F.: "A vehicle-to-vehicle communication protocol for cooperative collision warning". In: Proceedings of the 1st annual international conference on mobile and ubiquitous systems, networking and services, pp. 114–23. MOBIQUITOUS'04. IEEE (2004)
- Yu, J.Y., Chong, P.H.J.: A survey of clustering schemes for mobile ad hoc networks. In: Communications Surveys and Tutorials, vol. 7, no. 1, pp. 32–48. IEEE, First Qtr. (2005)
- Fernando, Niroshinie, Loke, Seng W., Rahayu, Wenny: Mobile cloud computing: a survey. Futur. Gen. Comput. Syst. **29**(1), 84–106 (2013)
- Bali, R.S., Kumar, N., Rodriguez, J.J.P.C.: Clustering in vehicular ad hoc networks: taxonomy, challenges and solutions. Veh Commun **1**(3), 134–152 (2014)

11. Yu, Rong, et al.: Toward cloud-based vehicular networks with efficient resource management. *Netw IEEE* **27.5**, 48–55 (2013)
12. Hamid Reza., A., Reza Ebrahimi, A., Saman, K.: FcVcA: a fuzzy clustering-based vehicular cloud architecture. In: *Communication Technologies for Vehicles (Nets4Cars-Fall)*, 2014 7th International Workshop, pp. 24–28. IEEE (2014)
13. Erfan, A., Hossein, R.: Expansion of vehicular cloud services on crossroads using fuzzy logic and genetic algorithm. In: *Computer Science and Software Engineering (JCSSE)*, 2015 12th International Joint Conference on. IEEE (2015)
14. Tian, D., Wang, Y., Lu, G., Yu, G.: A VANETs routing algorithm based on Euclidean distance clustering. In: *2nd IEEE International Conference on Future Computer and Communication*, Wuhan, pp. V1-183–V1-187 (2010)
15. Salhi, I., Cherif, M., Senouci, S.: Data collection in vehicular networks. *ASN symposium*. pp 20–21 (2008)
16. Chang, W., Lin, H., Chen, B.: "TrafficGather: an efficient and scalable data collection protocol for vehicular ad hoc networks". In: *5th IEEE Conference on Consumer Communications and Networking*, Las Vegas, NV, pp. 365–369 (2008)
17. Peng, F., et al.: Cluster-based framework in vehicular ad-hoc networks. *Ad-hoc, mobile, and wireless networks*, pp. 32–42. Springer, Berlin (2005)
18. Almalag, Mohammad S., Olariu, S., Weigle, M.C.: TDMA cluster-based mac for vanets (tc-mac). In: *World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2012 IEEE International Symposium on a IEEE (2012)
19. Kuzmanovic, A., Knightly, E.W.: TCP-LP: A distributed algorithm for low priority data transfer. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*. IEEE Societies, vol. 3. IEEE (2003)
20. Parameswaran, Ambili T., Husain, Mohammad I., Upadhyaya, S.: Is rssi a reliable parameter in sensor localization algorithms: an experimental study. In: *Field Failure Data Analysis Workshop (F2DA09)* (2009)
21. Ou, Chia-Ho: A roadside unit-based localization scheme for vehicular ad hoc networks. *Int. J. Commun. Syst.* **27**(1), 135–150 (2014)
22. McLachlan, Norman William: *Bessel functions for engineers*. Clarendon Press, Oxford (1961)
23. Website: <http://map.tehran.ir>, (2015)
24. Varga, A.: *OMNeT++ Discrete Event Simulation System User Manual*. 4.2.2. (2011)
25. Behrisch, M., Bieker, L., Erdmann, J., Krajzewicz, D.: SUMO—simulation of urban mobility: an overview. *SIMUL 2011, The Third International Conference on Advances in System Simulation*, pp. 63–68 (2011)
26. Sommer, C.: *Vehicles in network simulation (VEINS) Project*. Website: <http://veins.car2x.org>. (2012)
27. Al-Sultan, S., Moath, M., Al-Bayatti, Ali, Zedan, H.: A comprehensive survey on vehicular ad hoc network. *J. Netw. Comput. Appl.* **37**(1), 380–392 (2014)
28. Gerlough, Daniel L.: *Simulation of traffic flow*. Highway Research Board Special Report 79 (1964)
29. Fishman, G.S.: *Monte Carlo: Concepts, Algorithms, and Applications*. Springer Verlag, New York (1995)
30. Feller, W.: *An introduction to probability theory and its applications*, vol. I (1950)
31. Hassan, A., et al.: V-Cloud: vehicular cyber-physical systems and cloud computing. In: *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*. ACM (2011)