IEEE Open Journal of
Vehicular Technology

Received 11 August 2020; accepted 28 September 2020. Date of publication 5 October 2020;
date of current version 22 October 2020. The review of this article was coordinated by Editor S. Sun.

Digital Object Identifier 10.1109/OJVT.2020.3028684

The Delay-Constrained and Network-Situation-Aware V2V2I VANET Data Offloading Based on the Multi-Access Edge Computing (MEC) Architecture

CHUNG-MING HUANG (Senior Member, IEEE), AND CHI-FENG LAI

Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 701, Taiwan

CORRESPONDING AUTHOR: CHUNG-MING HUANG (e-mail: huangcm@locust.csie.ncku.edu.tw).

This work was supported by the Ministry of Science and Technology (MOST), Taiwan (R.O.C.) under Grant MOST 108-2221-E-006-056-MY3.

ABSTRACT This paper proposes a Multi-access Edge Computing (MEC)-based delay-constrained k-hop-limited VANET data offloading method. In the proposed method, a vehicle X can launch the vehicle-to-vehicle-to-infrastructure (V2V2I) VANET data offload from the cellular network to the IEEE 802.11p/802.11bd Road Side Unit (RSU) when (i) X is not in the corresponding RSU's signal coverage and (ii) there is a multiple-hop vehicle-to-vehicle (V2V) path connecting X with a vehicle Y that is inside the RSU's signal coverage. The MEC paradigm is used to check whether a k-hop-limited V2V2I offloading path exists for vehicle X or not based on the periodically reported contexts from vehicles. In addition to use the V2V2I path's lifetime and quality to derive the suitable V2V2I offloading paths using the snapshot VANET topology on the time point of receiving X's reported context, this work proposed a scheme to derive the potential V2V2I paths that may exist in the future, i.e., in a constrained time interval, to find the better V2V2I offloading path. The performance evaluation shown that the proposed method outperforms the one that (i) only considers the data offloading path's lifetime or (ii) does not consider the candidate V2V2I offloading paths existed in the future.

INDEX TERMS Multi-access Edge Computing (MEC), VANET Data Offloading, Delay-constrained Computing, Vehicle to Vehicle (V2V) and Vehicle to Vehicle to Infrastructure (V2V2I).

I. INTRODUCTION

More and more research for techniques and applications/services of vehicular network, which has been called Vehicle Ad Hoc Network (VANET), Internet Of Vehicle (IOV), Connected Vehicle, etc., are under working because of the increasing interest of autonomous vehicles and smart vehicles [1]–[3]. That is, many aspects of vehicular network's research results will become real and usable in our daily life over the next generation cellular network. There are currently two networking technologies that are considered for vehicular network. One is IEEE 802.11p that has been studied since more than 10 years ago, for which the technical scenario is called Dedicated Short Range Communication (DSRC) [4], and its follow-up standard IEEE 802.11bd, which is under defining [5]. The other one is cellular network-based, which has been studied recently and is normally called the Cellular Vehicle to X (C-V2X) in 4G [6], [7] and NR C-V2X in 5G [5]. It is expected that both technologies would co-exist in the next generation wireless mobile networking environment, i.e., in the 5G era. The practical example is the co-existence of Wi Fi network and cellular network since 2G, 2.5G, 3G, 3.5G and 4G, for which each one has its usage domain for users' convenience and thus both of them can co-exist smoothly [4].

VANET data offloading [8], which is called VANET offloading for simplicity hereafter, denotes to offload vehicular network's data traffic from the LTE cellular network to a Wi Fi AP, an IEEE 802.11p/802.11bd-based Road Side Unit (RSU) or a small Base Station (BS) of 4G or 5G cellular network when the networking environment and situation are allowed. The main benefit of data offloading is to reduce the

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

networking load in the LTE cellular network from the operator side's concern and get the higher bandwidth or even save the expense of using cellular network from the user side's concern. For simplicity, this work adopts the IEEE 802.11p RSU-based VANET offloading scenario. But, the proposed methodology of this work can also be applied to the Wi Fi, small BS and IEEE 802.11bd -based VANET offloading.

Two scenarios of VANET offloading are V2V offloading and V2I offloading. The traditional scenario of the V2I VANET offloading is as follows. Vehicle X can communicate with its peer entity in the Internet using IEEE 802.11p network when X is inside the signal coverage of an IEEE 802.11p RSU; when X is outside the signal coverage of the RSU, X communicates with its peer entity in the Internet using LTE cellular network. Some works have been done for VANET offloading [9]–[13], which adopted the scenario of 1-hop away RSU, i.e., it has the offloading when the vehicle is inside RSU's signal coverage, 2-hop away RSU, or the store-and-forward opportunistic contacting. The V2V VANET offloading is as follows [14]. Let two moving vehicles be communicating with each other. Normally, the peered two vehicles use 4G/5G cellular to communicate with each other. If there is a k-hop V2V VANET path that exists between these two peered vehicles, then they can switch to use the k-hop V2V path to communicate with each other until the k-hop V2V path is disappeared. This work will concentrate on the V2I VANET offloading scenario.

To have the higher utilization of IEEE 802.11p-based VANET data offloading, the following scenario, which is called V2V2I offloading, was proposed in [15]. Let there be a RSU ahead of a moving vehicle X. (i) Before vehicle X enters into its ahead RSU's signal coverage, X can enable data offloading using a n-hop V2V ad hoc path that connects X with an ahead vehicle Y, which is in the ahead RSU's signal coverage and thus is able to use a V2I link to connect with RSU. Vehicle Y is called offloading agent hereafter. Fig. 1(a) depicts the scenario of VANET data offloading with the ahead RSU using a n-hop V2V path. (ii) When vehicle X has left the RSU's signal coverage, X still can have the data offloading using a n-hop V2V ad hoc path that connects X with a rear vehicle Y, which is in the rear RSU's signal coverage and is able to use a V2I link to connect with RSU. Fig. 1(b) depicts the scenario of VANET data offloading with the rear RSU using a n-hop V2V path.

The key point of having the V2V2I VANET offloading is how to find a suitable V2V2I path between the source vehicle X and X's ahead RSU. In contrast to use the distributed computing way in the traditional VANET study, the Multi-access Edge Computing (MEC)-based architecture [16], [17] was adopted to have the centralized computing way to find the V2V2I path [15]. The functional scenario of the proposed MEC-based method is as follows. Let each BS be associated with an MEC server. All of the RSUs and vehicles that are inside BS W's signal coverage are administered by the corresponding MEC server of BS W. Each vehicle needs to report its context, including (i) its location, speed, and ID, (ii) IDs of its neighboring vehicles, for which a neighboring vehicle Y of a vehicle X means that X and Y are in mutual signal coverage and X/Y can receive Y's/X's hello messages sent from Y's/X's On-Board-Unit (OBU), (iii) IDs of sensed RSUs, etc., to the MEC server periodically. On the time point that the MEC server receives the context report from vehicle X, which plans to have the V2V2I offloading, the V2V2I path that has the longest path's lifetime is selected for X based on the snapshot VANET topology, which consists of the connected topology of the MEC server's administered vehicles and RSUs, on that time point.

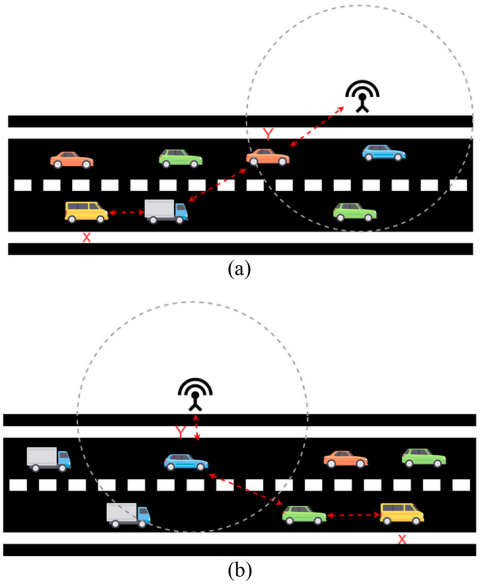

FIGURE 1. An example of the V2V2I VANET data offloading scenario.

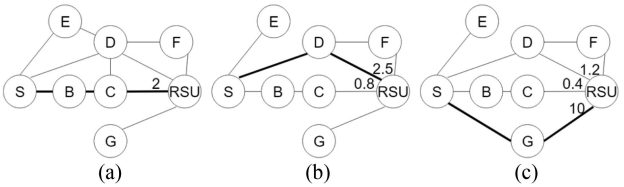

FIGURE 2. Illustrated utility-based vehicular ad hoc network topologies, (a) the initial one, (b) the one after 1 time unit, (c) the one after 2 time units.

Some concerns of the method proposed in [15] are as follows. The 1st concern is that the proposed method only considers the lifetime of the V2V2I path. Since different paths have different networking situations, it needs to use the utility, e.g., considering both path's lifetime and congestion level, to measure the quality of each candidate V2V2I VANET offloading path and then select the one that has the highest utility as the V2V2I VANET offloading path. The 2nd concern is that a V2V2I path that has the better utility may exist just some time later than the time point of receiving the context report. Referring to Fig. 2 as an example. Let the best V2V2I VANET offloading path based on the initial VANET topology that is snapshot on reference time point 0 be S-B-C-RSU and its utility is 2, which is depicted in Fig. 2(a). After one time

unit. links of C-D and D-E are ceased. Then, the best V2V2I path is S-D-RSU and its utility is 2.5, which is depicted in Fig. 2(b). Thereafter, after two time units, link S-G appears. In this condition, the best V2V2I VANET offloading path is S-G-RSU and its utility is 10, which is depicted in Fig. 2(c). That is, a better V2V2I offloading path can be derived if it waits for some time units.

Since it can't ensure the V2V2I VANVET offloading path that can exist at the current time point is a suitable one, the delay-constrained mechanism is adopted in this work to get the V2V2I VANVET offloading path. Let the current time point be T_c. The delay-constrained mechanism means that (i) the V2V2I paths that exist on the current time point T_c and (ii) those potential V2V2I paths that may exist in the following t time units, i.e., during the interval of $[T_c, T_c + t]$, are considered as the potential candidates for finding the better V2V2I VANET offloading path. That is, the MEC server selects the most suitable V2V2I VANET offloading path, which has the best utility value, based on the networking topologies that can exist during the interval of $[T_c, T_c + t]$.

Based on the aforementioned concerns, this work proposed the delay-constrained k-hop-limited V2V2I VANET offloading path's construction (*DC-KUPC*) method to tackle the V2V2I VANET offloading. Let a V2V2I offloading session denote the time period between (i) the time point that the source vehicle V_s starts to have V2V2I VANET offloading through the IEEE 802.11p RSU and (ii) the time point that the source vehicle V_s ends the V2V2I VANET data offloading and switches from the IEEE 802.11p network back to the cellular network. The goal of the proposed delay-constrained k-hop-limited V2V2I VANET offloading method is to maintain a V2V2I offloading session as long as possible. The principle of having a V2V2I offloading session is twofold: (1) While in the V2V2I offloading session, when the current V2V2I offloading path is broken, it tries to find the other V2V2I offloading path, whose hop count is normally (i) smaller than the previous one's hop count when the source vehicle is driving toward the ahead RSU and (ii) bigger than the previous one's hop count when the source vehicle is driving away the rear RSU more and more, to continue the V2V2I offloading session; (2) when the V2V2I offloading session is over, i.e., no V2V2I offloading path can be found, the source vehicle switches back to the cellular network. The finding of the next V2V2I offloading session is triggered by the receiving of the next reported context. Main contribution of this work are as follows: (1) Designing the VANET V2V2I path's construction method that takes delay-constrained time and networking condition into consideration and (2) designing the V2V2I VANET offloading method in which an offloading session is composed of several shrinking/extending V2V2I paths that are derived considering delay-constraint time and networking condition issues.

The rest of this paper is organized as follows: Section II presents related works for VANET offloading. Section III gives the functional scenario of the proposed method. Section IV introduces the delay-constrained k-hop-limited utility-based V2V2I VANET offloading path's construction method in detail. Section V shows the performance evaluation for the proposed method. Section VI gives the conclusion remarks.

II. RELATED WORK

In [9], the authors tackled the problem of the excessive data load for cellular network's BSs in the urban street intersections, which is caused by a lot of vehicles' simultaneously downloading data from cellular network. The authors proposed to use the offloading to solve the problem, for which the offloading is done when the vehicle is temporarily stop because of the red light. A vehicle downloads the data that are not sensitive to delay from a lightly loaded BS and becomes the data provider for other vehicles when it stops temporarily because of the red light.

In [10], the authors considered the scenario of offloading cellular traffic to RSUs. The proposed method offloaded data according to the popularity of data. The popular data is disseminated using RSUs and the unpopular data is disseminated using cellular network. Three scheduling methods were devised for different scenarios. The first one is based on individual RSU, in which each RSU caches the most popular data and broadcasts them. The second one is based on several RSUs whose signal coverage areas are overlapped. This method tries to find the best data allocation strategy to distribute the cached data in these overlapped RSUs. The third one is based on the movements of vehicles, in which the RSU predicts the each vehicle's movement to adjusts the cached data.

The aforementioned methods for data offloading considered the 1-hop scenario; in contrast, our work tries to extend the 1-hop VANET offloading to the multi-hop VANET offloading.

In [11], the authors designed the 2-hop offloading mechanism, in which the vehicle can offload data through the other vehicle that is in the signal coverage of the roadside Wi-Fi APs. The authors applied the M/G/1/K queueing process to analyze the effective service time of Wi-Fi and the helper vehicle's available probability. The experimental results shown that the proposed analysis model is very similar to the simulation result and thus it can be used in the access protocol designed for having the high mobility pattern. Comparing with our work, this work did not consider the more than 2-hop away data offloading's scenario.

In [12], the authors considered the scenario of the heavy load of the cellular network and proposed a cooperative downloading method in the heterogeneous vehicular network to relieve the congestion situation in the cellular network. In this work, the authors used three ways to offload data. (1) The vehicle directly downloads data using RSU by itself, i.e., using the self-offloading method when it is inside the signal coverage of the RSU. (2) The vehicle gets the data through the other vehicle that is inside RSU's signal coverage, i.e., it is the 2-hop away offloading based on our definition of the V2V2I offloading. (3) The vehicle X gets the data through the other vehicle Y when X and Y are contacting, i.e., when X and Y are in mutual signal coverage of their OBUs. It belongs to the carry-and-forward method in which Y carries data from the

remote RSU and then Y transmits the carried data to X when X and Y are contacting. The authors converted the problem into a graph based on each vehicle's mobility and throughput. To find the best strategy, the authors used the greedy method to optimize the loading time for video in each data flow.

In [13], the authors considered the scenario of data offloading using the V2V link that is affected by the high mobility of vehicles. The authors used the concept of the Vehicular Delay Tolerant Network (VDTN) to improve the efficiency of the offloading. The offloading is made through the central controller mechanism that can decide the data requests from vehicles are directly downloaded through the content server or downloaded through the VDTN. If the data requests can be downloaded through the VDTN, the central controller chooses some vehicles to download data through the content server and then let them disseminate data to the corresponding vehicles. The authors designed a mathematical framework by considering the contact duration of vehicles to deal with the mobility problem of vehicles. That is, the authors proposed the optimal resource allocation method by considering both contact duration and the limited buffer size of the vehicle.

The aforementioned works considered data offloading using the 1-hop or contacting data offloading scenario. In contrast, our work extends the data offloading to the multi-hop away data offloading scenario.

In [18], the authors considered the scenario of how to have data offloading through one of the available networks, e.g., Wi Fi, IEEE 802.11p, small cells, etc. The authors adopted the Software-Defined Network (SDN) and has a controller to make the decision of data offloading. The proposed method used the offload manager and the priority manager as the basis for the controller. The offload manager is used to manage the tasks of offloading. If the data volume is huge, the data should be offloaded. The priority manager is used to check the priority of the data to be offloaded. If the priority is low, the corresponding data could not be offloaded. The authors designed the efficient network allocator using the Stackelberg-Game theory to find the most appropriate network for offloading.

The aforementioned work mainly focused on how to dispatch the data offloading to the appropriate network and/or RSU; in contrast, our work mainly focused on how to maximize the data offloading performance through suitable k-hop-limited V2V2I VANET offloading paths.

III. THE FUNCTIONAL SCENARION OF THE PROPOSED METHOD

Let vehicle V_s be the vehicle that wants to have the IEEE 802.11p V2V2I VANET offloading and each vehicle will report its context to the MEC server of its connected BS periodically. Five phases of the proposed method are as follows: (i) the initialization phase: in this phase, vehicle V_s uses the LTE cellular network and the MEC server tries to find a V2V2I VANET offloading path for V_s, (ii) the shrinking phase: in this phase, vehicle V_s is driving toward the ahead RSU's signal coverage and the V2V2I VANET offloading path is shrinking gradually, i.e., the number of hops in the corresponding V2V2I path is becoming smaller gradually, (iii) the self-offloading phase: in this phase, vehicle V_s is inside RSU's signal coverage and can have the offloading by itself without the help of other vehicles, (iv) the extending phase: in this phase, vehicle V_s is driving away RSU's signal coverage more and more gradually, i.e., the number of hops in the corresponding V2V2I path is becoming bigger gradually and (v) the path recovery phase: in this phase, vehicle V_s's V2V2I VANET offloading path is broken unexpectedly and thus the path recovery mechanism is invoked to repair the corresponding V2V2I path.

A. THE INITIAL PHASE

Initially, vehicle V_s is using the 4G/5G cellular network to communicate with its peer V_p that is in the infrastructure Internet side. When the MEC server receives the periodically reported context from vehicle V_s on time point T_c, the MEC server is triggered to calculate whether there is one or more V2V2I offloading paths for vehicle V_s. When the MEC server finds that there is a V2V2I offloading path for V_s, the MEC server calculates all of the potential V2V2I paths that may exist during the interval of $[T_c, T_c + t]$ and select the best one as the V2V2I offloading path, which is generated on time point T_o, in which T_o is in the interval of $[T_c, T_c + t]$, for V_s. Then, the MEC server sends messages to the constituent vehicles of the corresponding V2V2I offloading path to enable the V2V2I VANET offloading session for V_s on time point T_o. The delay-constrained time length t is set as being smaller than or equal to the corresponding vehicle's periodically context reporting time period because the MEC server can receive the next reported context after one periodically reporting time period. During the offloading session, all of the constituent vehicles of the V2V2I offloading path report their contexts to the MEC server synchronously, which can be done through the use of aligning the time with GPS's time. After receiving the reported contexts of the constituent vehicles of the V2V2I offloading path, the MEC server calculates each V2V link time and the V2I link time, which denotes the offloading agent's staying time length inside RSU's signal coverage, to update the remaining offloading time of the corresponding V2V2I offloading path.

B. THE SHRINKING PHASE

The V2V2I offloading path must be kept as long as possible. Let the current time point that the MEC server receives the offloading agent's periodically reported context be T_c and the offloading agent will be out of RSU's signal coverage when its next periodically reported context is received, i.e., the offloading agent is leaving RSU's signal coverage. The shrinking algorithm is triggered to find the new offloading agent and the updated V2V2I offloading path, which is selected by the MEC server from all of the potential ones that may exist during the interval of $[T_c, T_c + t]$. Let the V2V2I offloading path's sub-path that is outside RSU's signal coverage be sub-$path_{out\text{-}RSU}$. Since it is unreasonable to have delay-constrained time length t to be longer than the lifetime of $sub\text{-}path_{out\text{-}RSU}$, after which

the V2V2I path is disappeared, the delay-constrained time length t is defined as being smaller than or equal to the lifetime of $sub\text{-}path_{out\text{-}RSU}$. If it cannot find a new offloading agent and/or the updated V2V2I offloading path, the MEC server will inform vehicle V_s to switch back to cellular network and the offloading session is ended. When vehicle V_s is driving inside RSU's signal coverage, the shrinking phase is ended and vehicle V_s has the offloading by itself.

C. THE SELF-OFFLOADING PHASE

When vehicle V_s is inside RSU's signal coverage, vehicle V_s can communicate with its V_p through RSU directly. In this phase, the MEC server keeps computing vehicle V_s's staying time length in RSU's signal coverage when it receives the periodically reported context from V_s. Let the current time point that the MEC server receives the periodically reported context of vehicle V_s be T_c and vehicle V_s will be out of RSU's coverage when its next periodically reported context is received, i.e., vehicle V_s is leaving RSU's signal coverage, it is changed to the extending phase.

D. THE EXTENDING PHASE

When the source vehicle V_s or the offloading agent of the extending phase is leaving RSU's signal coverage, the extending algorithm is triggered. Let the current time point that the MEC server receives the offloading agent's periodically reported context be T_c and the source vehicle or the offloading agent will be out of RSU's signal coverage when its next periodically reported context is received, i.e., the source vehicle or the offloading agent is leaving RSU's signal coverage. The extending algorithm tries to find the new offloading agent and the updated V2V2I offloading path, which is selected by the MEC server from all of the potential ones that may exist during the interval of $[T_c, T_c + t]$. Let the current V2V2I offloading path from vehicle V_s to offloading agent, which is inside and is leaving RSU's signal coverage, be $sub\text{-}path_{c\text{-}offloading}$. Since it is unreasonable to have delay-constrained time length t to be longer than the lifetime of $sub\text{-}path_{c\text{-}offloading}$, after which the V2V2I path is disappeared, the delay-constrained time length t is defined as being smaller than or equal to the lifetime of $sub\text{-}path_{c\text{-}offloading}$. If it cannot find a new offloading agent and/or the updated V2V2I offloading path, the MEC server will inform vehicle V_s to switch back to cellular network and the offloading session is ended.

E. THE PATH RECOVERY PHASE

When a V2V2I offloading path is broken unexpectedly, the path recovery algorithm is triggered to repair the V2V2I offloading path. Let $V_b - V_y - V_a$ be a sub-path in a V2V2I offloading path for V_s. When V_b can't transmit data to V_a, e.g., V_y left the road and thus the V2V2I offloading path is broken. Let the broken time point be T_c. The path recovery algorithm tries to find a new vehicle V_x that can repair the V2V2I offloading path or vehicle V_b will reconnect with V_a during the interval of $[T_c, T_c + t]$. Let the sub-path of the current offloading V2V2I offloading path from V_s to V_a be $sub\text{-}path_{s\text{-}a}$. Let the sub-path of the current offloading V2V2I offloading path from V_b to offloading agent be $sub\text{-}path_{b\text{-}OA}$. Since it is unreasonable to have delay-constrained time length t to be longer than the lifetime of $sub\text{-}path_{s\text{-}a}$ or $sub\text{-}path_{b\text{-}OA}$, after which the V2V2I offloading path is disappeared, the delay-constrained time length t is defined as being smaller than or equal to the minimum lifetime of $sub\text{-}path_{s\text{-}a}$ and $sub\text{-}path_{b\text{-}OA}$. If it cannot repair V2V2I offloading path, the MEC server will inform vehicle V_s to switch back to cellular network and the V2V2I offloading session is ended.

TABLE 1 The Variables that are Used for the Presentation of the Proposed Method

Symbol	Description
V_{Ref}	The given reference vehicle.
O	O contains all vehicles, but some vehicles are too far away to be a constituent vehicle for the V2V2I offloading path, the range of the considered vehicles is from V_{Ref} to RSU plus those vehicles that are on the rear of V_{Ref} and will drive into the range between V_{Ref} and RSU in the delay constraint time period.
P_x (p)	The position of object p in the x direction.
P_y (p)	The position of object p in the y direction.
V_x (p)	The velocity of object p in the x direction.
V_y (p)	The velocity of object p in the y direction.
$Link$ (p, q)	The link between object p and object q.
T_{start} $(Link\ (p, q))$	The time point that $Link\ (p,q)$ starts to exist, i.e., object p can start to communicate with object q from this time point.
T_{end} $(Link\ (p, q))$	The time point that $Link\ (p,q)$ is terminated, i.e., object p can't communicate with object q from this time point.
$Delay_{limit}$	The upper bound of the delay-constrained time length.
$Link_{limit}$	The upper bound of the end time point of the link.
T_{set}	The set of the delay time points to be considered
R	The signal coverage range
L_set	The link set that stores all of the links. At the beginning, it contains all of the links such that all of the considered objects in the object set $(O_1, O_2, ..., O_n)$ are fully connected.

IV. CONTROL SCHEMES FOR THE PROPOSED METHOD

In this Section, the proposed delay-constrained k-hop-limited utility-based V2V2I offloading path's construction (DC-KUPC) method is presented.

Table 1 depicts the variables that are used to present the proposed method. The DC-KUPC control scheme has four input variables: (i) the given reference vehicle V_{Ref}, (ii) the upper bound of the delay-constrained time length $Delay_{limit}$, (iii) the upper bound of the end time point of the link $Link_{limit}$, (iv) the upper bound of hop-count k.

The DC-KUPC control scheme has three main steps: (i) Find all of the links' connected time intervals: The goal of this Step is to find the links' start time points and end time points in the link set L_set. Let the time point that the MEC server received the reported context from the offloading source vehicle or the constituent vehicle in the offloading path and started

Procedure 1: Find All of the Links' Connected Time Intervals.

1: **for each** *Link (p, q)* in L_set **do**
2: *SolveLinkConnectTime*
3: **End for**

to execute the DC-KUPC control scheme be zero and be the reference time for the follow-up time counting. (ii) Generate the set of delay time points: The goal of this step is to generate the set of the delay time points to be considered, which is called T_{set} hereafter. Before adding delay time points to T_{set}, each link's connected time interval has to be adjusted to satisfy $Delay_{limit}$ and $Link_{limit}$. (iii) Exploring the best offloading path: The goal of this step is to find the best offloading path for the given vehicle V_{Ref} of the DC-KUPC control scheme. The algorithm uses *L_set* and T_{set} to generate vehicular ad hoc network topology at the corresponding delay time point dynamically.

A. THE INITIAL PHASE

The four input variables in the initial phase of the DC-KUPC are as follows: (i) the V_{Ref} is V_s in the initial phase, (ii) the $Delay_{limit}$ is the vehicle's periodically context reporting time period, (iii) the $Link_{limit}$ is infinity and (iv) the k is given.

The execution steps of the DC-KUPC control scheme are as follows.

1) FIND ALL OF THE LINKS' CONNECTED TIME INTERVALS

The first step is finding the start time points and end time points of all links. Procedure 1 is executed for Step 1, in which the *SolveLinkConnectTime* function is for deriving the start time point and end time point of each link.

Function *SolveLinkConnectTime* can use (i) the distance between a pair of object p and object q in *Link (p, q)* at time t and (ii) the range R that object p and object q can connect with and (ii) the range R that object p and object q can connect with each other, e.g., R is equal to 300m for the IEEE 802.11p OBU, to find the time points that object p and object q are at the boundary of the mutually connected area based on Equation (1), for which the derivation details using Equation (1) is presented in the Appendix.

$$\left[P_x(p) - P_x(q) + t\left(V_x(p) - V_x(q)\right)\right]^2 + \left[P_y(p) - P_y(q) + t\left(V_y(p) - V_y(q)\right)\right]^2 - R^2 = 0 \quad (1)$$

If the number of Equation (1)'s solution of t for *Link (p, q)* is zero, it means that the shortest distance between object p and object q is larger than R, then the *Link (p, q)* is removed from the *L_set.* If it has one solution, it means that *Link (p, q)* is broken immediately, then the link is removed from the *L_set.* If it has two solutions, then it means that the *Link (p, q)* exists for a time interval, in which one solution is the start time point and the other one is the end time point of *Link (p, q).* If it has the infinite number of solutions, it means that *Link (p, q)* can exist forever, i.e., they have the same driving speed and direction and the distance between them is not larger than R initially.

TABLE 2 The Variables That are Used in the Utility Function

Symbol	Description
$T^{offloading}$	The lifetime of the V2V2I offloading path, which is the minimum link's connected time of the constituent links in the V2V2I offloading path.
N_i	The number of vehicles that are in the signal coverage of the offloading path's constituent vehicle V_i.
N^{max}	Max (N_1, N_2, ..., N_n).
N^{total}	Sum of (N_1, N_2, ..., N_n).
n	The hop count in the V2V2I offloading path, including the V2I link from the offloading agent to RSU.

2) GENERATE THE SET OF DELAY TIME POINTS

In this Step, the proposed utility formula is presented and those generated links that satisfy the delay constraint are found and then valid delay time points are generated.

i) Offloading Path's Utility: A utility function needs to be devised to denote a candidate V2V2I offloading path's quality. The main considerations of deriving utility are as follows: (i) the longer lifetime a V2V2I path is, the larger utility value it can have, (ii) if there are more vehicles inside a constituent vehicle's signal coverage, then it results in more potential transmission collision, e.g., it needs the higher backoff value for getting the privilege of using the transmission channel, and thus it has the smaller utility value, (iii) if the sum of the number of vehicles inside each constituent vehicle's signal coverage is bigger, then it denotes that the V2V2I path is in a potentially more congested road, and thus it has the smaller utility value.

Based on the aforementioned considerations, the utility function is defined as follows. For convenient explanation, Table 2 depicts the variables that are used to define the utility function.

$$\text{Utility} = \frac{T^{\text{offloading}}}{\alpha\left(2^{\lfloor \log_2 N^{\max} \rfloor} + N^{\max}\right) + (1-\alpha)N^{\text{total}}} \cdot \frac{1}{\frac{\sqrt{\sum_{i=1}^{n} i^2}}{n}} \quad (2)$$

Since the constituent vehicles and RSU in the V2V2I offloading path transmit data continuously, the process of transmitting data in the V2V2I offloading path is hop by hop and the pipeline form. Thus, the transmission rate is limited by the most congested link in the V2V2I offloading path. The other factor is the congested situation of the whole path, which results from how the congested situation the corresponding road is. The transmission rate of the most congested link is affected by its contention window, for which the estimated

Procedure 2: Generate the set of Delay Time Points.

1: $T_{set} \leftarrow \emptyset$
2: **for each** *Link (p, q)* in *L_set* **do**
3: **if** T_{start} *(Link (p, q))* $<$ $Delay_{limit}$ **and** T_{end} *(Link (p, q))* > 0 **then**
4: T_{start} *(Link (p, q))* $\leftarrow$ *max* (T_{start} *(Link (p, q)), 0)*
5: T_{end} *(Link (p, q))* $\leftarrow$ *min* (T_{end} *(Link (p, q)),* $Link_{limit}$*)*
6: *add* T_{start} *(Link (p, q))* *to* T_{set}
7: **if** T_{end} *(Link (p, q))* $<$ $Delay_{limit}$ **then**
8: add T_{end} *(Link (p, q))* to T_{set}
9: **else**
10: remove *Link (p, q)* from *L_set*

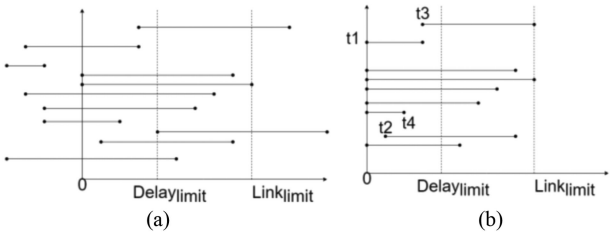

FIGURE 3. An illustrated example of the timelines' distribution of all generated links in L_set, (a) the original one and (b) the modified one after executing Procedure 2.

minimum number of collisions is $\lceil \log_2 N^{\max} \rceil$. The estimated waiting time slot is in proportion to $2\lfloor^{\log_2 N^{\max}}\rfloor + N^{\max}$, in which N^{max} denotes the number of the estimated waiting time slots that needs to be experienced for the last contention, i.e., for the successful transmission. N^{total} denotes the congestion index for the whole V2V2I offloading path, for which the bigger N^{total} is the more congested situation it has. For $\frac{1}{\sqrt{\frac{\sum_{i=1}^{n} i^2}{n}}}$, in which n denotes the number of links, including the V2I link connecting the offloading agent and RSU, in the V2V2I offloading path, it denotes the stability of the n-hop V2V2I offloading path. Since the longer V2V2I offloading path is, i.e., n is bigger, the smaller value $\frac{1}{\sqrt{\frac{\sum_{i=1}^{n} i^2}{n}}}$ is, i.e., it denotes the V2V2I path having the higher probability to be broken.

ii) Generate the Set of Delay Time Points: Procedure 2 is executed for finding those generated links that satisfy the delay constraint and then generating valid delay time periods.

For convenient explanation, Fig. 3 depicts an illustrated example. The distribution of the timeline that each generated link exists is depicted in Fig. 3(a). Fig. 3(b) shows the distribution of timelines of the valid links after the checking of satisfying the delay constraint. The principles are follows: (i) The links that are disconnected before time zero are removed. (ii) The links that are connected after $Delay_{limit}$ are removed. (iii) The remaining links are those valid links whose original/modified T_{start} is not smaller than 0.

Lines 6-8 of Procedure 2 are used for the processing of the second Step, which puts the starting time point T_{start} and the ending time point T_{end} of the valid links into the T_{set} set. Referring to Fig. 3 as an example, T_{set} contains *t1, t2, t3, t4* which represent that the corresponding vehicular ad hoc network topology is changed on time points *t1 − t4*. That is, a new vehicular ad hoc network topology is generated when it is snapshot on each of the time points *t1 − t4 b*ecause a new link is generated or a link is ceased, which results in the utility of each potentially existed V2V2I offloading path being changed, and thus a better V2V2I offloading path may exist potentially. In other viewpoint, each time point in T_{set} denotes a new vehicular ad hoc network topology that can be derived in the delay-constrained interval. As a result, the MEC server can select the V2V2I offloading path that has the highest utility from all of the potentially existed V2V2I offloading paths derived in all of the potentially existed vehicular ad hoc network topologies that are snapshot on time points *t1, t2, t3, t4*.

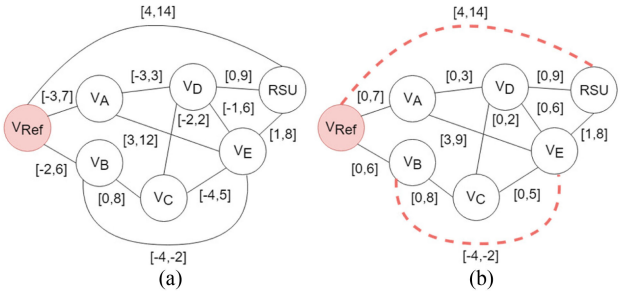

FIGURE 4. The illustrated topology for Fig. 3.

TABLE 3 The Variables That Are Used in the Exploring Step

Symbol	Description
Path	The path from V_{Ref} to *Tail (Path)* at *Delay (Path).*
Tail (Path)	The ID of the vehicle or RSU of the Path.
Delay (Path)	The delay time of the Path.
Extend_from_Node (Path)	The ID of the vehicle that is connected with *Tail (path).*
Hop (Path)	The hop's count of the Path.
LT (Path)	The lifetime of the Path.
Utility (Path)	The utility of the Path.
Candidate_set	The set of the candidate offloading paths.

Fig. 4 is an illustrated topology for Fig. 3. Fig. 4(a) contains all of the originally generated links; Fig. 4(b) depicts the modified topology, in which the removed links are $V_{Ref} - RSU$ and V_B - V_E, and some links' start time points are adjusted to zero, the end time point of a link, i.e., $V_A - V_E$, is adjusted to $Link_{limit}$, which is 9, and the aforementioned time points *t1, t2, t3, t4* of Fig. 3 are 0, 1, 3, 2 respectively.

3) Exploring the Best Offloading Path: This Step is to explore the potential V2V2I offloading path link by link, until the best V2V2I offloading path is found. Table 3 depicts the

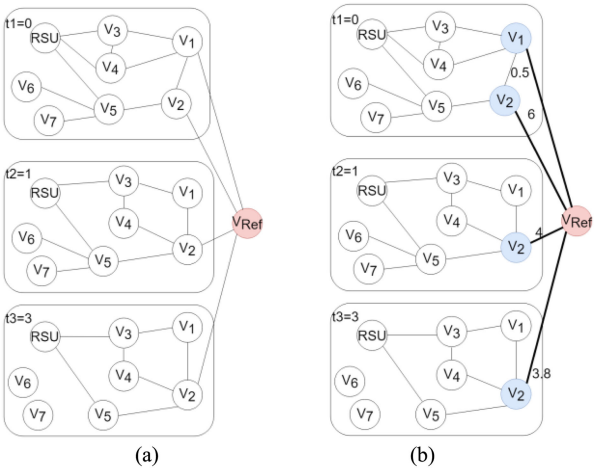

FIGURE 5. **(a) The illustrated example for Algorithm 1; (b) the illustrated example of executing the InitCandidateSet function.**

Algorithm 1: *Exploring the Utility-Based Offloading Path* (V_{Ref}, T_{set}, k).

```
1:  Candidate_set ← InitCandidateSet (V_Ref, T_set)
2:  while Candidate_set is not empty do
3:    select the exploring path Candidate_Path_max
4:      that has the maximum utility from
        Candidate_set
5:    if Tail (Candidate_Path_max) = RSU then
6:      break
7:    UpdateCandidateSet(Candidate_set,
      Candidate_Path_max, k)
8:    remove Candidate_Path_max from Candidate_set
9:  return V2V2I_offloading_path
    (Candidate_Path_max)
```

variables that are used in this Step. Algorithm 1 is executed for Step 3.

Algorithm 1 is explained as follows, for which the illustrated topology depicted in Fig. 5(a) is used for the convenient explanation. Line 1 of Algorithm 1 uses the *InitCandidateSet* function to generate the initial *Candidate_se*t, which contains all of the one-hop path from V_{Ref}. Lines 2-8 of Algorithm 1 extends the candidate paths by exploring the *Candidate_set*. If the *Candidate_set* is not empty and RSU has not been reached, the candidate paths will continuously be extended. During exploring the *Candidate_set*, the candidate path that has the highest utility among those in *Candidate_set* is chosen to be explored and is called *Candidate_Path*$_{max}$.

Referring to Fig. 5(a) as an example. Let the valid delay time points be *t1, t2, t3,* and the corresponding delay time lengths are 0, 1, 3 respectively. Each node denotes an object, which denotes a vehicle or the RSU, and a line exists between two objects that can communicate with each other directly. The node with the red color means that it has been visited and can't be explored again. At the beginning, Algorithm 1 is started from reference vehicle V_{Ref}. Thus, it is marked as the red color. The *InitCandidateSet* function explores the links that are originated from the reference vehicle V_{Ref}.

Function *InitCandidateSet* (V_{Ref}, T_{set})

```
1:  Candidate_set ← ∅
2:  for each v in O do
3:    for each t in T_set do
4:      if Link (V_Ref, v) exists and T_start (Link (V_Ref,
        v)) <= t and T_end (Link (V_Ref, v)) > t then
5:        Tail (new_candidiate_path) ← v
6:        Delay (new_candidiate_path) ← t
7:        Extend_from_Node (new_candidiate_path)
          ← V_Ref
8:        LT (new_candidiate_path) ← T_end (Link
          (V_Ref, v)) – t
9:        Hop (new_candidiate_path) ← 1
10:       Utility (new_candidiate_path) ← Using
          Eq. 2 to solve
11:       add new_candidiate_path to Candidate_set
12: return Candidate_set
```

In the *InitCandidateSet* function, Line 4 finds those links that connect with the reference vehicle V_{Ref} in each vehicular ad hoc network topology existing in each valid delay time point t of T_{set}, for which the corresponding link's start time point should be smaller than or equal to delay time point t and the link's end time point should be greater than the delay time point t. Lines 5-11 add each qualified candidate one-hop path from V_{Ref} to *Candidate_set,* for which (i) the new candidate path's lifetime is the end time point minus the delay time point, (ii) the new candidate path's hop count is one because this is the first hop from V_{Ref}, (iii) the new candidate path's utility can use utility formula to get it. Fig. 5(b) depicts an example of executing the *InitCandidateSet*, in which (i) a blue node denotes the tail node of a candidate path, which is from V_{Ref}, in the *Candidate_set*, (ii) bold lines are those candidate paths in *Candidate_set,* (iii) the number associated with a link denotes the utility of the corresponding candidate path.

In Algorithm 1, after the initial candidate paths, whose hop count is 1, originated from the reference vehicle V_{Ref} are found and stored in *Candidate_set*, it can start to explore to find the best V2V2I offloading path. On line 7 of Algorithm 1, the *UpdateCandidateSet* function explores the links that start from vehicle *Tail (Candidate_Path*$_{max}$*)*.

In the *UpdateCandidateSet* function, Line 3 selects the one, i.e., *Candidate_Path*$_{max}$, that has the maximum utility from those candidate paths in *Candidate_set.* If the tail object of the *Candidate_Path*$_{max}$ is not RSU, which means that the path still needs to be extended in order to reach RSU. Lines 5-6 of the *UpdateCandidateSet* function explore those links that are ori–ginated from the tail of *Candidate_Path*$_{max}$ and whose (i) start time is earlier than the delay constraint time point and (ii) end time is after the delay constraint time point. Lines 6-12 of the *UpdateCandidateSet* function creates each new candidate path as follows: (i) record

that the *new_candidate_path* is extended from the tail of the *Candidate_Path*$_{max}$, (ii) the *new_candidate_path*'s lifetime is the minimum of the *Candidate_Path*$_{max}$'s lifetime and the exploring link's connected time length, because the whole path's lifetime is limited by the link that has the shortest lifetime, (iii) the *new_candidate_path*'s hop count is *Candidate_Path*$_{max}$'s hop count plus 1 because it is the next hop from *Candidate_Path*$_{max}$, (iv) the *new_candidate_path*'s utility can be derived using the utility formula depicted in Eq. 2. Lines 13-17 of the *UpdateCandidateSet* function check whether the tail object of the *new_candidate_path* has been explored by the other candidate path or not: if it has not, then adding the *new_candidate_path* to the *Candidate_set*; if it has, then compare the utility of these two candidate paths. If the new candidate path's utility is higher than the existed one, then replace the existed one with the new one in *Candidate_set*.

Function UpdateCandidateSet(Candidate_set,Candidate_Path$_{max}$*, k)*

```
1:  if Hop (Candidate_Path_max) < k then
2:      t ← Delay (Candidate_Path_max)
3:      Tail_vehicle ← Tail (Candidate_Path_max)
4:      for each v in O do
5:          if Link (Tail_vehicle, v) exists in T_set and T_start
            (Link (Tail_vehicle, v)) <= t and T_end (Link
            (Tail_vehicle, v)) > t then
6:              Tail (new_candidate_path) ← v
7:              Delay (new_candidate_path) ← t
8:              Extend_from_Node(new_candidiate_path)
                ←Tail_vehicle
9:              LT(new_candidate_path)←
                min(LT(Candidate_Path_max),T_end(Link
                (Tail_vehicle, v)) – t)
10:             Hop (new_candidate_path) ← Hop
                (Candidate_Path_max) + 1
11:             Utility (new_candidate_path) ← Using Eq. 2
                to solve
12:             if v is the tail of the other candidate path w
                in Candidate_set then
13:             if Utility (new_candidate_path) > Utility (w)
                then
14:                 replace w with new_candidate_path
15:             else
16:                 add new_candidate_path to Candidate_set
```

Figs. 6–8 depict the example of exploring the candidate paths. In this example, the limit of the hop count, which is called k-limited, is 3.

Referring to Fig. 6(a), the one that has the maximum utility candidate path is V_{Ref} - V_2 on delay time point *t1*. Since the tail is V_2, which is not RSU, it needs to explore from V_2. After executing the *UpdateCandidateSet* function, the *new_candidate_path* V_{Ref} - V_2 - V_5, whose utility is 1.5, is added into *Candidate_set*. The other *new_candidate_path* V_{Ref} - V_2 - V_1 's utility, which is 3.9, is larger than the

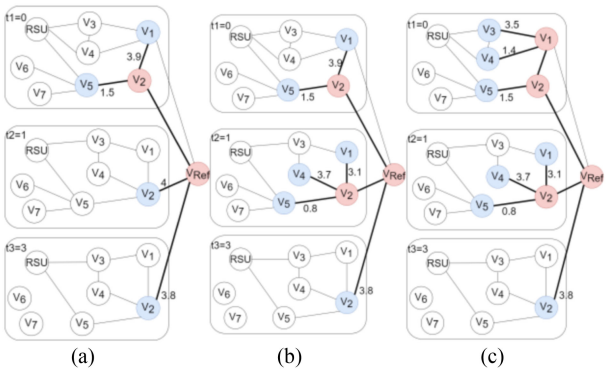

FIGURE 6. The example of exploring of the candidate paths – part 1.

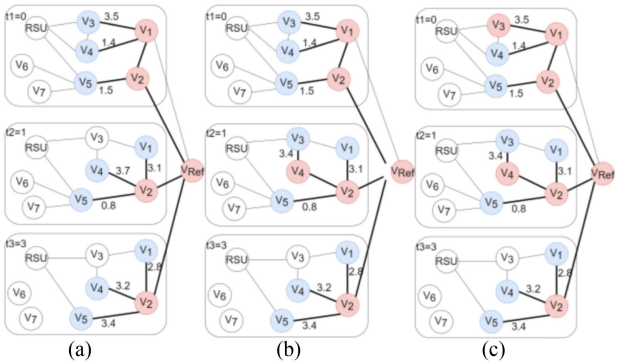

FIGURE 7. The example of exploring of the candidate paths – part 2.

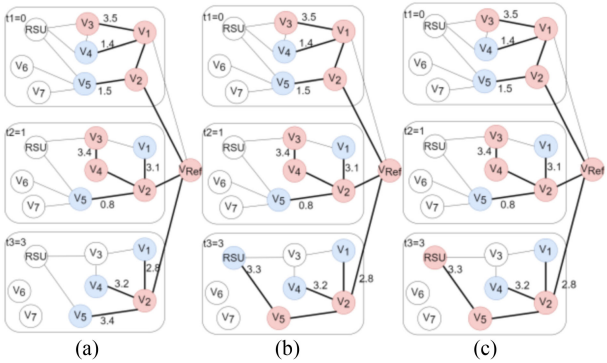

FIGURE 8. The example of exploring of the candidate paths – part 3.

utility of the older candidate path V_{Ref} - V_1. Thus, the old one is replaced by the new one in *Candidate_set*. Referring to Fig. 6(b), the one that has the maximum utility candidate path is V_{Ref} - V_2 on delay time point *t2*. Since the tail is V_2, which is not RSU, it needs to explore from V_2. After executing the *UpdateCandidateSet* function, the *new_candidate_path* V_{Ref} - V_2 - V_5, whose utility is 0.8, is added into *Candidate_set*. The other *new_candidate_path* V_{Ref} - V_2 - V_4, whose utility is 3.7, is also added into *Candidate_set*. The other *new_candidate_path* V_{Ref} - V_2 - V_1, whose utility is 3.1, is also added into *Candidate_set*. Referring to Fig. 6(c), the one that has the maximum utility candidate path is V_{Ref} - V_2 - V_1 on delay time point t1. Since the tail is V_1, which is not RSU,

it needs to explore from V_1. After executing the *UpdateCandidateSet* function, the *new_candidate_path* V_{Ref} - V_2 - V_1 - V_4, whose utility is 1.4, is added into *Candidate_set*. The other *new_candidate_path* V_{Ref} - V_2 - V_1 - V_3, whose utility is 3.5, is also added into *Candidate_set.*

Referring to Fig. 7(a), the one that has the maximum utility candidate path is V_{Ref} - V_2 on delay time point *t3*. Since the tail is V_2, which is not RSU, it needs to explore from V_2. After executing the *UpdateCandidateSet* function, the *new_candidate_path* V_{Ref} - V_2 - V_1, whose utility is 2.8, is added into *Candidate_set*. The other *new_candidate_path* V_{Ref} - V_2 - V_4, whose utility is 3.2, is also added into *Candidate_set*;the other *new_candidate_path* V_{Ref} - V_2 - V_5, whose utility is 3.4, is also added into *Candidate_set*. Referring to Fig. 7(b), the one that has the maximum utility candidate path is V_{Ref} - V_2 - V_4 on delay time point *t2*. Since the tail is V_4, which is not RSU, it needs to explore from V_4. After executing the *UpdateCandidateSet* function, the *new_candidate_path* V_{Ref} - V_2 - V_4 - V_3, whose utility is 3.4, is added into *Candidate_set*. Referring to Fig. 7(c), the one that has the maximum utility candidate path is V_{Ref} - V_2 - V_1 - V_3 on delay time point *t1*. Since the tail is V_3, which is not RSU, it needs to explore from V_3. Nevertheless, since the hop count of V_{Ref} - V_2 - V_1 - V_3 is equal to the hop limit k, i.e. 3, the *UpdateCandidateSet* function cannot be executed.

Referring to Fig. 8(a), the one that has the maximum utility candidate path is V_{Ref} - V_2 - V_4 - V_3 delay time point *t2*. Since the tail is V_3, which is not RSU, it needs to explore from V_3. Nevertheless, since the hop count of V_{Ref} - V_2 - V_4 - V_3 is equal to the hop limit k, i.e. 3, the *UpdateCandidateSet* function cannot be executed. Referring to Fig. 8(b), the one that has the maximum utility candidate path is V_{Ref} - V_2 - V_5 on delay time point *t3*. Since the tail is V_5, which is not RSU, it needs to explore from V_5. After executing the *UpdateCandidateSet* function, the *new_candidate_path* V_{Ref} - V_2 - V_5 - *RSU*, whose utility is 3.3, is added into *Candidate_set*. Referring to Fig. 8(c), the one that has the maximum utility candidate path is V_{Ref} - V_2 - V_5 - *RSU* on delay time point *t3*. Since the tail is RSU, the offloading path is found and the algorithm is terminated.

B. THE SHRINKING PHASE

The shrinking phase is triggered on the time point that the MEC server receives the offloading agent's periodically reported context and the offloading agent will be out of RSU's signal coverage when its next periodically reported context is received. Then, let the time point of receiving the reported context be zero and be the reference time point for the follow-up time counting. Fig. 9 depicts an illustrated example for convenient explanation. In the shrinking phase, the four input variables of the DC-KUPC control scheme are as follows: (i) The reference vehicle V_{Ref} is the constituent vehicle that is closest to RSU and isn't in the signal coverage of RSU of the V2V2I offloading path. Referring to Fig. 9, V_z is the V_{Ref}. (ii) The $Delay_{limit}$ is set as the lifetime of the sub-path from V_s to V_{Ref}, i.e., V_z. The reason is that the whole V2V2I offloading path from V_s to RSU may be disappeared/ended if the delay time length is longer than the lifetime of the sub-path. (iii) The $Link_{limit}$ is the lifetime of the sub-path from V_s to V_{Ref}, i.e., V_z. Due to the whole offloading path's lifetime is the minimum connected time length of all links in the path, the lifetime of the sub-path from V_s to RSU has to be limited by the lifetime of the sub-path from V_s to V_{Ref}. (iv) The hop count limit k is given. After executing the *DC-KUPC* control scheme, the new sub-path from V_{Ref} to RSU is found. Then the modified V2V2I offloading path consists of the sub-path from V_s to V_{Ref} and the sub-path from V_{Ref} to RSU.

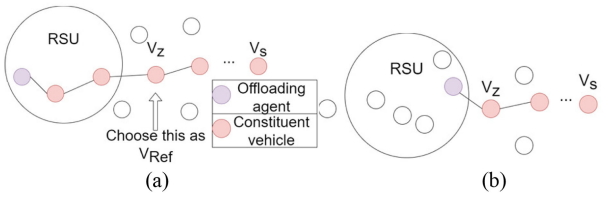

FIGURE 9. The illustrated configurations (a) before and (b) after executing the shrinking phase.

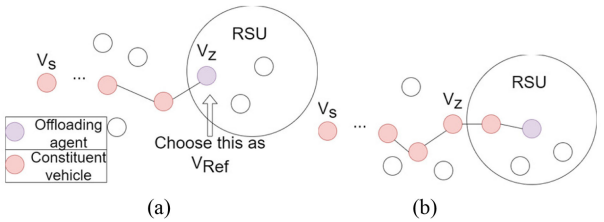

FIGURE 10. The illustrated configurations (a) before and (b) after executing the extending phase.

C. THE EXTENDING PHASE

The extending phase is triggered on the time point that the MEC server receives the offloading agent's periodically reported context and the offloading agent will be out of RSU's signal coverage when its next periodically reported context is received. Then, let the time point of receiving the reported context be zero and be the reference time point for the follow-up time counting. Fig. 10 depicts an illustrated example for convenient explanation. In the extending phase, the four input variables of the *DC-KUPC* control scheme are as follows: (i) The reference vehicle V_{Ref} is the constituent vehicle that is closest to RSU and is in the signal coverage of RSU of the V2V2I offloading path. Referring to Fig. 10, V_z is the V_{Ref}. (ii) The $Delay_{limit}$ is set as the lifetime of the sub-path from V_s to V_{Ref}, i.e., V_z. The reason is that the whole V2V2I offloading path from V_s to RSU may be disappeared/ended if the delay time length is longer than the lifetime of the sub-path. (iii) The $Link_{limit}$ is the lifetime of the sub-path from V_s to V_{Ref}, i.e., V_z. Due to the whole offloading path's lifetime is the minimum connected time length of all links in the path, the lifetime of the sub-path from V_s to RSU has to be limited by the lifetime of the sub-path from V_s to V_{Ref}. (iv) The hop count limit k is given. After executing the *DC-KUPC* control scheme, the new sub-path from V_{Ref} to RSU is found. Then the modified

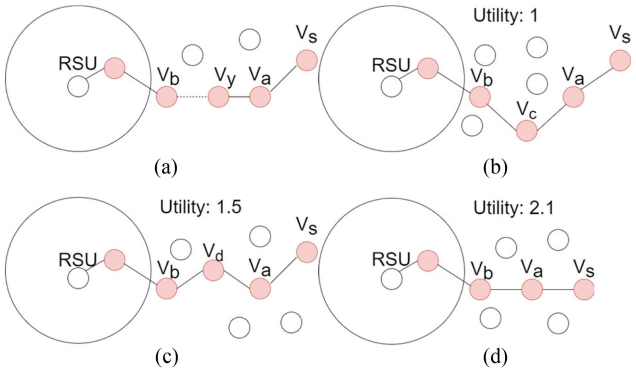

FIGURE 11. The example of updating the time of links of which the offloading path is composed, (a) the initial one, (b) the one after delaying 1 time unit, (c) the one after delaying 2 time units and (d) the one after delaying 3 time units.

V2V2I offloading path consists of the sub-path from V_s to V_{Ref} and the sub-path from V_{Ref} to RSU.

D. THE PATH RECOVERY PHASE

Fig. 11 depicts an illustrated example for convenient explanation. The purpose of the path recovery phase is to try to connect the two sub-paths, in which one is from V_s to V_a and the other one is from V_b to RSU depicted in Fig. 11, by one vehicle or directly, i.e., by a 1-hop's or two hops' sub-path.

The Path Recovery algorithm is similar to the *DC-KUPC* control scheme. It adopts Procedure 1 and Procedure 2 of the *DC-KUPC* control scheme to generate T_{set}. Then it creates the new offloading path by replacing or removing the disappeared vehicle with the other vehicle in the original V2V2I offloading path on the delay time points in T_{set}. It compares the lifetime of the new offloading path with that of the current best offloading path: if it is larger than that of the current best offloading path's lifetime, then the current best offloading path is replaced by the new one.

Referring to Fig. 11 as an example. Fig. 11(a) is the original offloading path and vehicle V_y is the disappeared vehicle. Fig. 11(b) depicts a new offloading path that is derived by replacing V_y with V_c on the occasion of delaying 1 time unit and the new offloading path's utility is 1. Fig. 11(c) is the other new offloading path that is derived by replacing V_y with V_d on the occasion of delaying 2 time units and the new offloading path's utility is 1.5. Fig. 11(d) depicts a new offloading path, whose utility is 2.1, by removing V_y after delaying 3 time units. Since it has the largest utility, the path recovery phase uses this offloading path to recover.

E. HANDOFF PROCESSING BETWEEN TWO MEC SERVERS

When a vehicle V_x in a V2V2I path is leaving BS_x 's signal coverage and entering into BS_y's signal coverage, it can handoff from BS_x to BS_y. Since each BS is associated with an MEC server, the MEC server handoff process needs to be triggered when BS's handoff happens.

Some assumptions are as follows. If a RSU is located in the overlapped region of two or more BSs, it can be administered by all of the corresponding BSs' MEC servers. MEC servers of those BSs whose signal coverages are overlapped need to exchange the reported contexts received from the vehicles and RSUs that are inside the corresponding BS's signal coverage. Let the process for launching and administering the V2V2I offloading session be handled by the master MEC server, in which the corresponding RSU is inside the signal coverage of the MEC server's associated BS. Related processing of the MEC server handoff for the proposed k-hop-limited V2V2I VANET offloading is as follows.

At the initial offloading phase, the source vehicle that is having the V2V2I offloading and the associated RSU are located in the same MEC server's region. The MEC server is this V2V2I offloading session's master server. Two conditions are as follows. (1) Let the RSU be in an individual BS's signal coverage. In this condition, the source vehicle or some constituent vehicles in the V2V2I offloading path may leave the master MEC server's administered region, i.e., they handoff to the neighboring BS, in the extending phase. In this condition, the associated neighboring MEC server would transmit the corresponding vehicle's reported contexts to the master MEC server and the V2V2I VANET offloading session can be maintained. (2) Let the RSU be in the overlapped region of two BSs' signal coverage. In this condition, the source vehicle or some constituent vehicles in the V2V2I offloading path may leave the master MEC server's administered region, i.e., they handoff to the neighboring BS, in the shrinking/extending phase. If the source vehicle that is having the V2V2I offloading handoff to the neighboring BS, then the master MEC server also needs to handoff to the neighboring BS's MEC server. Since the previous and the new master MEC servers are neighboring ones, the reported contexts of those constituent vehicles that are still in the administered region of the previous master MEC server can be forwarded to the new MEC server. Thus, the V2V2I VANET offloading session can be maintained.

If the number of the source vehicle's BS/MEC server handoff is more than one, the V2V2I offloading session is ended. The reason is that the V2V2I offloading path is too long from the source vehicle to the corresponded RSU, it will lead the path to be unstable.

V. PERFORMANCE EVALUATION

In this work, the network environment is simulated using NS3 and the mobility patterns are generated using SUMO. Table 4 lists the parameters used in the simulation environment. The vehicle drives in the 4.8 km × 20 m straight road with six intersections. Vehicle departure's period is set as 2s/1.25s/0.8s in the low/middle/high vehicle density's situation respectively. Vehicles departure from either end of the straight road and some vehicles may turn in the intersections. When a vehicle is driving through the road, it reports its context to the MEC server for (i) every 10 seconds regularly and (ii) every 5 seconds when it becomes a constituent vehicle of a V2V2I VANET offloading path and randomly transmits data through V2V links and downloads data from BSs. The data flow is

TABLE 4 The Parameters Used in the Simulation Environment

Parameter	Value
The vehicle departure period in the low/middle/high vehicle density's situation	2s/1.25s/0.8s
Vehicle speed limit	60 km/h
The vehicle's context reporting period when the vehicle is/isn't in the offloading path	5s/10s
RSU signal coverage	300m

from the peer entity that is in the infrastructure side to either (i) the BS of the LTE cellular network or (ii) the RSU of the IEEE 802.11p network to the source vehicle. Data packets are transmitted using the UDP protocol.

The compared offloading methods, i.e., the offloading path's construction strategies, are as follows: (1) the self RSU offloading, i.e., the offloading is enabled when the source vehicle is inside RSU's signal coverage, (2) the k-hop-limited lifetime-based offloading without path recovery, (3) the delay-constrained k-hop-limited lifetime-based offloading without path recovery, (4) the k-hop-limited lifetime-based offloading with path recovery, which is the proposed method in [15], (5) the delay-constrained k-hop-limited lifetime-based offloading with path recovery, (6) the k-hop-limited utility-based offloading without path recovery, (7) the delay-constrained k-hop utility-based offloading without path recovery, (8) the k-hop-limited utility-based offloading with path recovery, (9) the delay-constrained k-hop-limited utility-based offloading with path recovery.

The adopted performance matrices are as follows: (1) the total data offloading fraction is equal to the data that are transmitted through the RSU divided by the data that are transmitted through both BS and RSU, (2) the data loss rate is equal to the lost data, which are caused by (i) the collision when data is transmitting and (ii) the on-the-fly data that are dropped when the offloading path is broken, in the V2V2I offloading path divided by the data that are transmitted through RSU, (3) the successful data offloading fraction is equal to the data that are correctly received by the source vehicle through the V2V2I offloading path divided by the data that are transmitted through both BS and RSU, and (4) the total offloading time, which denotes the time that the source vehicle used the V2V2I offloading path to offload data during its driving through the road.

The comparison of the total data offloading fraction is depicted in Fig. 12. The total data offloading fraction is analyzed based on the following four factors.

i) The first factor is vehicle density. The total data offloading fraction in the low/middle vehicle density's situation is lower than that of the middle/high vehicle density's situation. The reason is that the higher vehicle density it is, the more chance it has to construct the V2V2I offloading path.

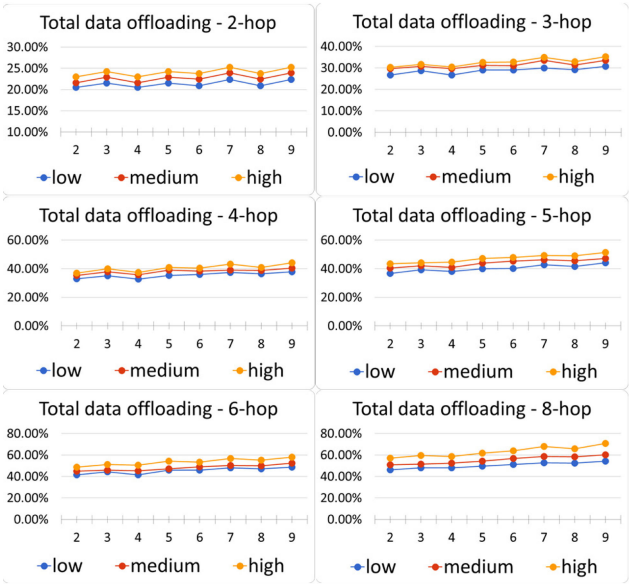

FIGURE 12. The comparison of the total data offloading fraction.

ii) The second factor is the limit of hop count. When the limit of the hop count is higher, the total data offloading fraction is also higher. The reason is that the limit of the hop count is higher, the length of the offloading path is longer; thus, the source vehicle can launch the offloading session earlier and terminate the offloading session later, which results in the number of offloading sessions being bigger and the lifetime of an offloading session being longer. Consequently, the total data offloading fraction becomes higher.

iii) The third factor is the delay constraint computing mechanism. The total data offloading fraction is higher when the MEC server has adopted the delay constraint computing mechanism. The advantages of using the delay constraint computing mechanism are as follows: (1) Some short lifetime offloading paths can be used. The reason is that if the lifetime is shorter than the vehicle's context reporting time period, the MEC server can't discover the offloading path without using the delay constraint computing mechanism. (2) The offloading path can be launched earlier. The reason is that the MEC server's calculating for the existence of the offloading path using the delay constraint computing mechanism can oversee the future V2V2I offloading paths; thus the corresponding path's offloading can be pre-set to be launched when they are available, i.e., without needing to wait for the next context reporting time point. That is, some of the offloading path's lifetime is wasted because of waiting to the vehicle's next context reporting time when the delay constraint computing mechanism is not used.

iv) The fourth factor is the path recovery mechanism. The total data offloading fraction is slightly higher when the path recovery mechanism is used, especially in the situations of the higher limit of hop count's situation.

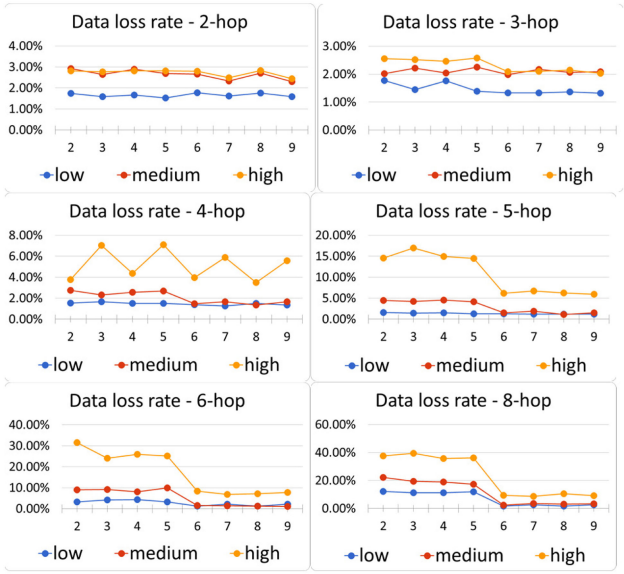

FIGURE 13. The comparison of the data loss rate.

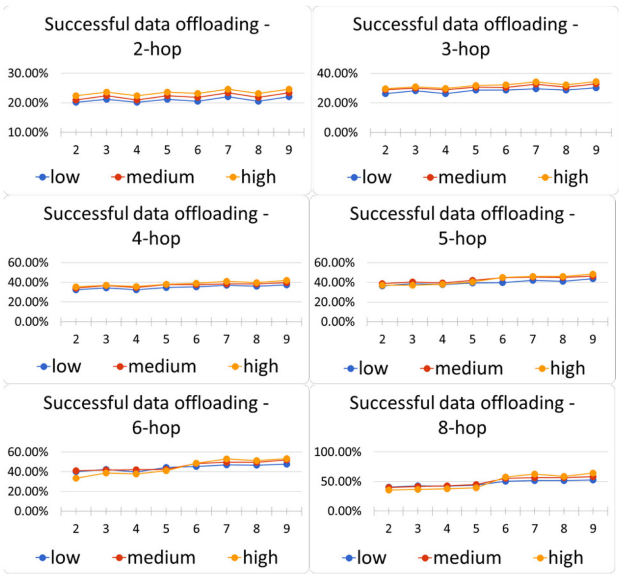

FIGURE 14. The comparison of the successful data offloading fraction.

The reason is that the path recovery mechanism is able to find a vehicle to repair the path to extend the lifetime of the offloading session. It is more effective in the situation of the higher limit of hop count because the corresponding paths are much easier to find substituent vehicles to repair the paths.

Additionally, comparing the lifetime-based method and the utility-based method, the total data offloading fraction is slightly higher when the utility-based method is used. The reason is that the utility-based method tends to use the lower hop count's offloading paths whose mobility is more stable than that of the higher hop count's offloading paths. Thus, the utility-based method's total data offloading fraction is higher than that of the lifetime-based method, especially in the higher limit of hop count's situation.

The comparison of the data loss rate is depicted in Fig. 13. The data loss rate is analyzed based on the following three factors.

i) The first factor is vehicle density. The data loss rate in the middle vehicle density's situation is generally (a) lower than that of the high vehicle density's situation and is (b) higher than that of the low vehicle density's situation. The reason is that the higher vehicle density leads to more collision when data is transmitting through the V2V2I offloading path. Thus, the data loss rate becomes higher

ii) The second factor is the limit of hop count. When the limit of hop count is higher, the data loss rate is also higher. Since the data transmitted through the V2V2I offloading path have to contend in each constituent link of the path, the more hop the offloading path is, the more contention the data experiences. Thus, the data loss rate is higher when the limit of hop count is higher. This effect is stronger when the vehicle density's situation is higher, i.e., the data loss rate in the low vehicle density's situation increases slower than that of the high vehicle density's situation when the limit of hop count increases.

iii) The third factor is using the lifetime-based or the utility-based method. The data loss rate is lower when the MEC server uses the utility-based method. The reason is that the utility-based method considers the V2V2I offloading path's congestion situation. The V2V2I offloading paths chosen by the utility-based method are less congested than the V2V2I offloading paths chosen by the lifetime-based method. The utility-based method is more effective in the middle and high vehicle density's situations because the difference of V2V2I offloading path's congestion level between the V2V2I offloading paths is higher in the middle and high vehicle density's situations

Additionally, it is observed that the data loss rate has (1) the sharp increment when the hop count is changed from 4 to 5 in the high vehicle density's situation using the lifetime-based method, (2) the big increment when the hop count is changed from 6 to 8 in the low/middle vehicle density's situation using the lifetime-based method, (3) the regular increment when the hop count increases from 2 to 8 in the high vehicle density's situation using the utility-based method and (4) the more or less stable condition when the hop count increases from 2 to 8 in the low/middle vehicle density situation using the utility-based method. Nevertheless, it needs to combine both factors of the data loss rate and the total data offloading volume to have the more effective observation, for which the successful data offloading fraction is used and is presented in the following part.

The successful data offloading fraction is used to judge the effectiveness of different offloading methods. The successful data offloading fraction is affected by both the data loss rate and the data offloading fraction. Referring to Fig. 14, the

successful data offloading fraction keeps increasing when the limit of hop count increases in the low/middle/high vehicle density's situation using the lifetime-based method. But, when the limit of hop count increases, the low/middle vehicle density's situation gradually becomes better than that of the middle/high vehicle density's situation. The reason is that when the total data offloading fraction becomes higher, the data loss rate is increased. Unfortunately, the increment rate of the data loss rate is higher than that of the total data offloading fraction. Thus, the successful data offloading fraction becomes worse when the vehicle density's situation becomes higher and the limit of the hop count becomes more. Additionally, the successful data offloading fraction becomes more stable (1) when the hop count reaches 4 in the high vehicle density's situation using the lifetime-based method and (2) when the hop count reaches 6 in the low/middle vehicle density's situation using the lifetime-based method.

Referring to Fig. 14, the successful data offloading fraction keeps increasing when the limit of hop count increases in the low/middle/high vehicle density's situation using the utility-based method. Additionally, the successful data offloading fraction of the high/middle vehicle density's situation is also always better than that of the middle/low vehicle density's situation. The reason is that, although the data loss rate is increased, the total data offloading fraction becomes higher too. Additionally, the increment rate of the data loss rate is lower than that of the total data offloading fraction. Thus, the successful data offloading fraction becomes better when the vehicle density's situation becomes higher and the limit of hop count becomes more.

Referring to Fig. 14, the successful data offloading fraction using the utility-based method is higher than the successful data offloading fraction using the lifetime-based method. The reason is that the utility-based method tries to select the better path and thus the data loss rate is lower than that of the lifetime-based method. Additionally, the successful data offloading fraction keeps increasing in the low, middle and high vehicle density's situations using the utility-based method; in contrast, the successful data offloading fraction is not increased (i) when the hop count reaches 6 in the low and middle vehicle density's situations and (ii) when the hop count reaches 4 in the high vehicle density's situation using the lifetime-based method.

The comparison with the traditional method, i.e., the self RSU offloading method, and the proposed methods is depicted in Fig. 15. Referring to Fig. 15, comparing with the best proposed method, i.e., the 8-hop-limited delay-constrained utility-based method with path recovery, the successful data offloading fraction of the proposed method outperforms the traditional method; even if using the worst proposed method, i.e., the 2-hop-limited utility-based method without path recovery, the successful data offloading fraction of the proposed method is still higher than that of the traditional method.

The comparison of the total offloading time is depicted in Fig. 16. Referring to Fig. 12 and Fig. 16, clearly, the total offloading time and the total data offloading fraction have

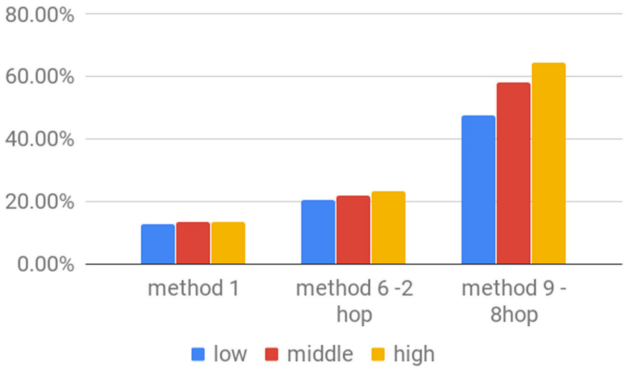

FIGURE 15. The comparison of the successful data offloading fraction using the traditional method and the proposed method.

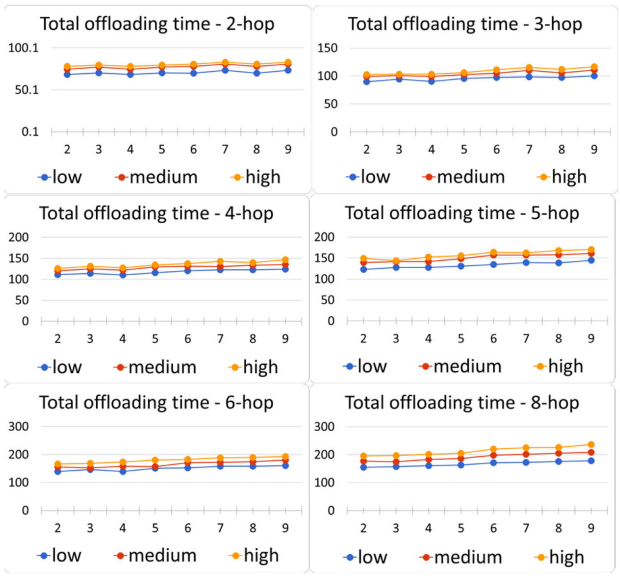

FIGURE 16. The comparison of the total offloading time.

significantly positive correlation, i.e., they have the same trends among the compared methods. Since the explanations were given previously, it is not repeated over here.

VI. CONCLUSION

This paper has proposed the MEC-based delay-constrained and network-situation-aware V2V2I VANET offloading method. This method can find a suitable k-hop-limited V2V2I VANET path for data offloading through the help of the MEC server. The delay-constrained mechanism means that (i) the V2V2I paths that exist on the current time point T_c and (ii) those potential V2V2I paths that may exist in the following t time units, i.e., during the interval of $[T_c, T_c + t]$, are considered as the potential candidates for finding the better V2V2I VANET offloading path. That is, the MEC server can take the delay-constrained time period into consideration to find the most suitable V2V2I VANET offloading path that has the highest utility value based on the networking topologies that can exist during the interval of $[T_c, T_c + t]$. Additionally, the networking condition, which is based on the contention

level, is also considered. Thus, an offloading session in the proposed V2V2I VANET offloading method is composed of several shrinking/extending V2V2I paths that are derived considering delay-constraint time t and networking condition issues. Based on the performance analysis, it can be found that the delay constrained computing mechanism extends the data offloading time because of the resulted more offloading sessions and longer offloading session time; the path recovery mechanism extends the data offloading time because of the resulted longer offloading session time. The performance analysis has shown that the proposed utility-based method outperforms (i) the lifetime-based method, which only considers the path's lifetime without considering the networking quality and (ii) the traditional self-offloading method, i.e., the offloading is enabled when the source vehicle is inside RSU's signal coverage in terms of the total data offloading fraction, the data loss rate and the successful data offloading fraction. The lifetime-based method can have 3.4 times of the successful data offloading fraction comparing with the traditional method. But the lifetime-based method's successful data offloading fraction can't be higher (1) when the limit of hop count reaches 4 in the high vehicle density situation and (2) when the hop count reaches 6 in the low/middle vehicle density situation. The proposed utility-based method can have 4 times of the successful data offloading fraction comparing with the traditional method and its successful data offloading fraction still increases when the limit of hop count increases. In this paper, only the one RSU's situation is considered. For future work, how to construct the better data offloading path in the situation of multiple RSUs and how to keep the offloading session when the handoff from one RSU to the other RSU happens are the problems to be considered.

APPENDIX

Let the time for snapshotting a VANET/MANET topology be 0. The derivation of the time period for two moving objects in the corresponding snapshotted VANET/MANET to be able to connect and communicate with each other is as follows.

Let the position of an object W in the x- and y- coordinate be represented as $Pos_x\ (W)$ and $Pos_y\ (W)$ respectively. The distance between a pair of object p and object q in Link(p, q) at time t can be represented as follows:

$$D\left(\text{Link}\left(\text{p},\text{q}\right),\ \text{t}\right) = \sqrt{\begin{array}{l}\left[Pos_x\left(p\right)-Pos_x\left(q\right)+t\left(Vel_x\left(p\right)-Vel_x\left(q\right)\right)\right]^2 \\ +\left[Pos_y\left(p\right)-Pos_y\left(q\right)+t\left(Vel_y\left(p\right)-Vel_y\left(q\right)\right)\right]^2\end{array}} \tag{A1}$$

If object p and object q can connect and communicate with each other in range R, e.g., R is equal to 300 m for the IEEE 802.11p OBU, then it can use R to find the time point that object p and object q are at the boundary of the mutually connected area. Thus, Equation (A1) becomes

$$\left[Pos_x\left(p\right)-Pos_x\left(q\right)+t\left(Vel_x\left(p\right)-Vel_x\left(q\right)\right)\right]^2 + \left[Pos_y\left(p\right)-Pos_y\left(q\right)+t\left(Vel_y\left(p\right)-Vel_y\left(q\right)\right)\right]^2 - R^2 = 0 \tag{A2}$$

Define

$$\overrightarrow{D_{\text{diff}}} = \begin{bmatrix} Pos_x\left(p\right)-Pos_x\left(q\right) \\ Pos_y\left(p\right)-Pos_y\left(q\right)\end{bmatrix}$$

$$\overrightarrow{D_{\text{diff}}} = \begin{bmatrix} Vel_x\left(p\right)-Vel_x\left(q\right) \\ Vel_y\left(p\right)-Vel_y\left(q\right)\end{bmatrix}$$

$$\overrightarrow{D_{\text{diff}}}^T = \left[\,Pos_x\left(p\right)-Pos_x\left(q\right)\ Pos_y\left(p\right)-Pos_y\left(q\right)\,\right]$$

$$\overrightarrow{D_{\text{diff}}}^T = \left[\,Vel_x\left(p\right)-Vel_x\left(q\right)\ Vel_y\left(p\right)-Vel_y\left(q\right)\,\right]$$

Thus, Equation (A2) can be rewritten as follows:

$$\left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}}\right)t^2 + 2\left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}}\right)t + \left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}} - R^2\right) = 0 \tag{A3}$$

Equation (A3) is a quadratic equation and the solution is shown in Equation (A4), shown at the bottom of the page.

There are three cases for having solutions of Equation (A4):

Case 1: $[2(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}})]^2 - 4(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}})(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}} - R^2) > 0$

In this Case, t has two solutions of real number, in which one denotes the start time point $(T_{start}\ (Link\ (x, y)))$, shown at the bottom of the next page, and the other one denotes the end time point $(T_{end}\ (Link\ (p, q)))$, shown at the bottom of the next page, that object p and object q can communicate with each other.

The corresponding movements and relative positions for objects p and q are depicted in Fig. 17.

Fig. 17(a) shows that object p is approaching object q and the distance between object p and object q is larger than R before time point T_{start}. Thus, object p and object q can't communicate with each other.

Fig. 17(b) shows that the distance between object p and object q is equal to R on time point T_{start}. Thus, object p and object q can communicate with each other from now on.

$$t = \frac{-2\left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}}\right) \pm \sqrt{\left[2\left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}}\right)\right]^2 - 4\left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}}\right)\left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}} - R^2\right)}}{2\left(\overrightarrow{D_{\text{diff}}}^T \times \overrightarrow{D_{\text{diff}}}\right)} \tag{A4}$$

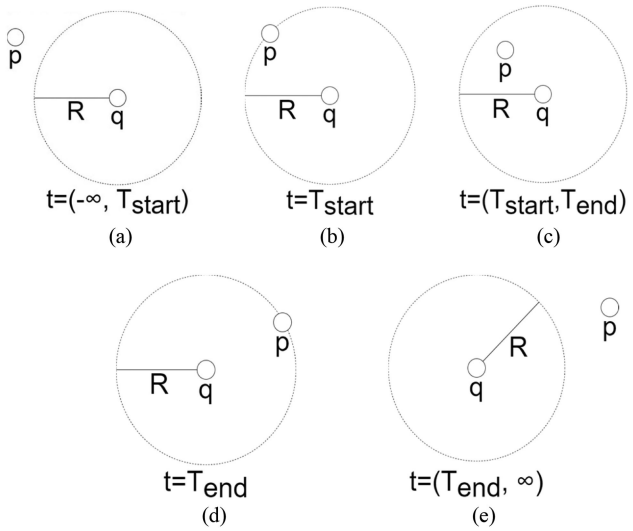

FIGURE 17. The corresponding movements and relative positions for objects *p* and *q* in Case 1.

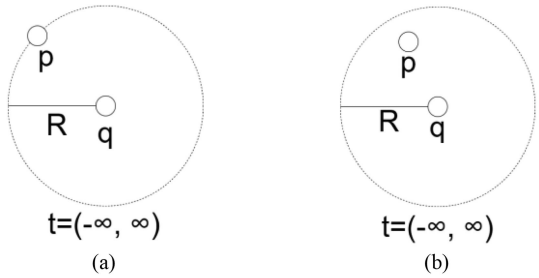

FIGURE 18. The corresponding movements and relative positions for objects *p* and *q* in Case 2- (i), (a) the relative distance is equal to *R*, (b) the relative distance is smaller than *R*.

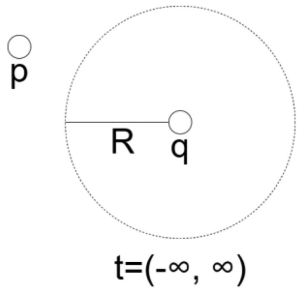

FIGURE 19. The corresponding movements and relative positions for objects *p* and *q* in case 2- (ii).

Fig. 17(c) shows that the distance between object p and object q is smaller than R between two time points T_{start} and point T_{end}. Thus, object p and object q can communicate with each other.

Fig. 17(d) shows that the distance between object p and object q is equal to R again on time point T_{end}. Thus, it is the last moment that object p and object q can communicate with each other.

Fig. 17(e) shows that object p is leaving away object q more and more and the distance between object p and object q is larger than R after time point T_{end}. Thus, object p and object q can't communicate with each other. Thus, *Link(p, q)* can be kept in *L_set*.

Case 2: $[2(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}})]^2 - 4(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}})(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}} - R^2) = 0$

In this Case, the number of t's solutions is zero, one or infinite and thus it can be further divided into three types

i) If $\overrightarrow{D_{\text{diff}}} = 0$ and $\overrightarrow{D_{\text{diff}}} \leq R$, then t has an infinite number of solutions. In this case, object p and object q can always connect with each other, i.e., they have the same speed and driving direction and thus their relative position is not changed, which condition can be depicted in Fig. 18(a) or Fig. 18(b). Thus, *Link (p, q)* can be kept in *L_set*.

ii) If $\overrightarrow{D_{\text{diff}}} = 0$ and $\overrightarrow{D_{\text{diff}}} > R$, then t has no solution. In this Case, object p and object q always can't connect with each other, i.e., they have the same speed and driving direction and thus their relative position is not changed, which is depicted in Fig. 19. Thus, *Link (p, q)* is removed from *L_set*.

iii) If $\overrightarrow{D_{\text{diff}}} \neq 0$, then it means that object p and object q connect and disconnect at the same time. In this situation, it is equal to always disconnect. Thus, *Link (p, q)* is removed from *L_set*. The corresponding movement and the relative position for object p and object q are depicted in Fig. 20.

Fig. 20(a) shows that object p is approaching object q and the distance between object p and object q is larger than R.

$$T_{start}\,(Link\,(p,q)) = \frac{-2\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right) - \sqrt{\left[2\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right)\right]^2 - 4\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right)\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}} - R^2\right)}}{2\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right)}$$

$$T_{end}\,(Link\,(p,q)) = \frac{-2\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right) + \sqrt{\left[2\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right)\right]^2 - 4\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right)\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}} - R^2\right)}}{2\left(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}}\right)}$$

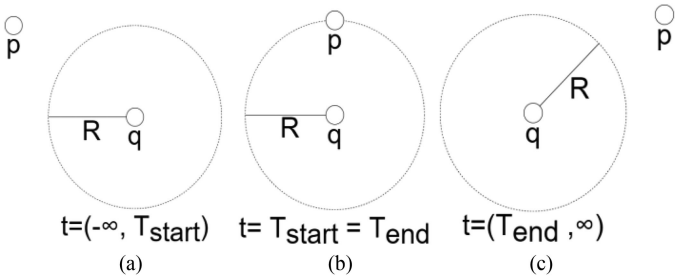

FIGURE 20. The corresponding movements and relative positions for objects *p* and *q* in Case 2- (iii).

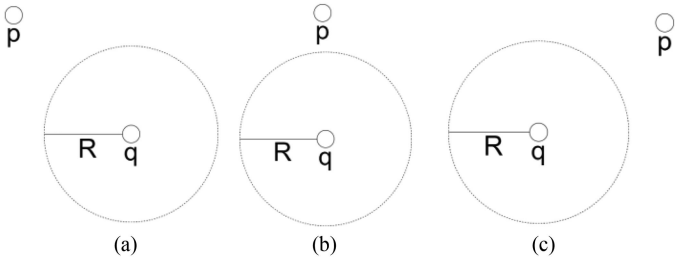

FIGURE 21. The corresponding movements and relative positions for objects *p* and *q* in Case 3.

Thus, object p and object q can't communicate with each other.

Fig. 20(b) shows that the shortest distance between object p and object q is R.

Fig. 20(c) shows that object p is leaving away object q more and more and object p can't communicate with object q.

Case 3: $[2(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}})]^2 - 4(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}})(\overrightarrow{D_{\text{diff}}}^{T} \times \overrightarrow{D_{\text{diff}}} - R^2) < 0$

In this Case, object p and object q always disconnect because there is no solution of real t. The corresponding movements and the relative positions for object p and object q are depicted in Fig. 21. Thus, *Link (p, q)* is removed from *L_set*. Fig. 21(a) shows that object x is approaching object y. Fig. 21(b) shows that the shortest distance between object x and object y is longer than R. Fig. 21(c) shows that object x is leaving away object y more and more.

REFERENCES

[1] Z. MacHardy, A. Khan, K. Obana, and S. Iwashina, "V2X access technologies: Regulation, research, and remaining challenges," *IEEE Commun. Surv. Tut.*, vol. 20, no. 3, pp. 1858–1877, Jul.–Sep. 2018.

[2] L. Ang, K. P. Seng, G. K. Ijemaru, and A. M. Zungeru, "Deployment of IoV for smart cities: Applications, architecture, and challenges," *IEEE Access*, vol. 7, pp. 6473–6492, 2019.

[3] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscape—Architectures, enabling technologies, applications, and development areas," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2391–2406, Aug. 2018.

[4] K. Abboud, H. A. Omar, and W. Zhuang, "Interworking of DSRC and cellular network technologies for V2X communications: A survey," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9457–9470, Dec. 2016.

[5] G. Naik, B. Choudhury, and J. M. Park, "IEEE 802.11bd & 5G NR V2X: Evolution of radio access technologies for V2X communications," *IEEE Access*, vol. 7, pp. 70169–70184, 2019.

[6] H. Seo, K. Lee, S. Yasukawa, Y. Peng, and P. Sartori, "LTE evolution for Vehicle-to-Everything services," *IEEE Commun. Mag.*, vol. 54, no. 6, pp. 22–28, Jun. 2016.

[7] S. Chen *et al.*, "Vehicle-to-Everything (v2x) services supported by LTE-based systems and 5G," *IEEE Commun. Standards Mag.*, vol. 1, no. 2, pp. 70–76, 2017.

[8] H. Zhou, H. Wang, X. Chen, X. Li, and S. Xu, "Data offloading techniques through vehicular ad hoc networks: A survey," *IEEE Access*, vol. 6, pp. 65250–65259, 2018.

[9] J. Feng and Z. Feng, "A vehicle-assisted offloading scheme for hotspot base stations on metropolitan streets," in *Proc. 28th IEEE Annu. Int. Symp. Personal, Indoor Mobile Radio Commun.*, 2017, pp. 1–6.

[10] T.-Y. Yu, X. Zhu, H. Chen, and M. Maheswaran, "HetCast: Cooperative data delivery on cellular and road side network," in *Proc. 28th IEEE Annu. Int. Symp. Personal, Indoor Mobile Radio Commun.*, 2017, pp. 1–6.

[11] X. Zhu, Y. Li, D. Jin, and J. Lu, "Contact-aware optimal resource allocation for mobile data offloading in opportunistic vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7384–7399, Aug. 2017.

[12] Y. Sun, L. Xu, Y. Tang, and W. Zhuang, "Traffic offloading for online video service in vehicular networks: A cooperative approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7630–7642, Aug. 2018.

[13] X. Zhu, Y. Li, D. Jin, and J. Lu, "Contact-aware optimal resource allocation for mobile data offloading in opportunistic vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7384–7399, Aug. 2017.

[14] C.-M. Huang, M.-S. Chiang, D.-T. Dao, W.-L. Su, S. Xu, and H. Zhou, "V2V data offloading for cellular network based on the software defined network (SDN) inside mobile edge computing (MEC) architecture," *IEEE Access*, vol. 6, pp. 17741–17755, 2018.

[15] C.-M. Huang, S.-Y. Lin, and Z.-Y. Wu, "The k-hop-limited V2V2I VANET data offloading using the mobile edge computing (MEC) mechanism," *Veh. Commun.*, vol. 26, 2020, Art. no. 100268.

[16] H. Liu, F. Eldarrat, H. Alqahtani, A. Reznik, X. d. Foy, and Y. Zhang, "Mobile edge cloud system: Architectures, challenges, and approaches," *IEEE Syst. J.*, vol. 12, no. 3, pp. 2495–2508, Sep. 2018.

[17] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tut.*, vol. 19, no. 4, pp. 2322–2358, Oct. /Dec. 2017.

[18] G. S. Aujla, R. Chaudhary, N. Kumar, J. J. Rodrigues, and A. Vinel, "Data offloading in 5G-enabled software-defined vehicular networks: A Stackelberg-game-based approach," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 100–108, Aug. 2017.

CHUNG-MING HUANG (Senior Member, IEEE) received the B.S. degree in electrical engineering from National Taiwan University, on June 1984, and the M.S. and Ph.D. degrees in computer and information science from The Ohio State University, on December 1988 and June 1991, respectively. He is currently a Distinguished Professor with the Department of Computer Science and Information Engineering (CSIE), National Cheng Kung University (NCKU), Tainan, Taiwan. He was a Chair of Department of CSIE and Director of Institute of Medical Informatics, NCKU and was the Chair of IEEE Vehicular Technology Society Tainan Chapter. He has authored or coauthored more than 350 referred journal and conference papers in wireless and mobile communication protocols, interactive multimedia systems, audio and video streaming and formal modeling of communication protocols. His research interests include wireless and mobile network protocol design and analysis, green computing and communication, media processing and streaming, and innovative network applications and services. He is a senior member of ACM.

CHI-FENG LAI received the B.S. degree in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan, on June 2017 and the master's degree in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan, on August 2019. His research interests include Vehicular Ad hoc Network.