Research Article
A Content Dissemination Technique Based on Priority to Improve Quality of Service of Vehicular Ad Hoc Networks

Muhammad Ayaz, Jawaid Iqbal, Muhammad Adnan, Saddam S. Hussain, Amerah Alabrah, Noor Ull Amin, Suheer Al-Hadhrami, and Sk. Md. Mizanur Rahman

1Department of Information Technology, Hazara University, Mansehra, Pakistan
2Department of Computer Science, Capital University of Science and Technology, Islamabad, Pakistan
3Department of Computer Science, University of Swabi, Swabi, Pakistan
4Department of Computer Science, Central South University, Changsha, China
5Department of Information Systems, College of Computer and Information Sciences, Riyadh 11543, Saudi Arabia
6Computer Engineering Department, Engineering College, Hadhramout University, Hadhramout, Yemen
7Information and Communication Engineering Technology, School of Engineering Technology and Applied Science, Centennial College, Toronto, Canada

Correspondence should be addressed to Suheer Al-Hadhrami; s.alhadhrami@hu.edu.ye

Received 20 January 2022; Revised 26 February 2022; Accepted 12 March 2022; Published 12 April 2022

Academic Editor: Muhammad Arif

Copyright © 2022 Muhammad Ayaz et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Due to the ability of Vehicular Ad Hoc Networks (VANETs) to improve mobility and create innovative services, they have received increasing attention from both industry and academia fields. Named Data Networking (NDN) is a network service that has been developed for Internet’s host-based packet delivery model. In VANETs, NDN is used as an emerging architecture for improving the quality of service (QoS) of the networks connected cloud and mobile Internet of Things (IoT). We can achieve this improvement using an efficient content dissemination and forwarding technique that depends on name-based routing, in-network content caching, and interest-based content retrieval. The large amount of data processed by several IoT sensors or nodes of VANETs makes content prioritizing and forwarding mechanism crucial, especially when many end-users need common content simultaneously. Therefore, in this paper, we propose a content dissemination technique based on priority for boosting the service requirements of NDN-based cloud and mobile IoT nodes in the VANETs. The approach prioritizes the data flow of traffic into four classes: urgent, emergency, least, and average using a novel constrained location and deadline method, called New method employing Deadline Distance and Size of Data (NDDS). The highest priority is assigned to the data traffic with the emergency class, then urgent, average, and least class. To evaluate this proposed technique, a numerical simulation experiment is conducted by using the CloudSim toolkit. The simulation results demonstrate that the emergency data flow has minimum processing time, compared to the average, urgent, and least data flow, which can significantly enhance the QoS of NDN protocol-based Ad Hoc networks.

1. Introduction

The rapid growth of Internet traffic has increased the need for researchers and academia to develop a new Internet architecture for the future that will address existing problems and issues in IP-based internets such as security, scalability, and high latency, which causes long delays and packet loss during mobility, among others. A new Internet architecture known as Information-Centric Networking (ICN) has been introduced to address these challenges [1]. ICN replaces “IP layer” with “contents”; it stores and distributes these contents using the router’s cache. Different architectures have been proposed in the literature, including Data-Oriented Network Architecture (DONA), Content-Centric Networking (CCN), Publish-Subscribe Internet Routing Paradigm (PSIRP), Network of Information
This research work is based on NDN architecture. Following is the overview of NDN architecture.

NDN is a promising network architecture for the Internet of the future. From host-centric to data-centric communications, there has been significant growth and evolution. NDN is more concerned with “what is data packet rather than where is a data packet,” which indicates that NDN is more concerned with data than with source node addresses [2, 3]. In NDN, every piece of information in a network has a name, which is used for routing, and data is stored in a cache in the router for better content distribution. These alternatives have a number of advantages, including decoupling sender and receiver dependencies, reducing network overhead, adaptive forwarding, and so on. As a result, NDN is the most cost-effective solution for VANET.

The rapid improvement and development in terms of automation, communication, storage devices, and computation have initiated a lot of significant and sensational services in VANET. VANET is a fundamental component of the Intelligent Transportation System (ITS), which has got a lot of interest from both industry and academicians in recent decades [4]. VANET provides a variety of applications, such as safety and infotainment applications, that benefit both the driver and the users. These applications and services are delivered through a variety of communication models, including Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communications [5, 6]. The VANET’s communication technology is the Dedicated Short Range Communication (DSRC) protocol and the Wireless Access in Vehicular Environment (WAVE) protocol, via which it is eventually operated. The TCP/IP protocol is at the heart of these protocol suites, enabling end-to-end communication. Apart from these many VANET architectures in literature have been proposed over Internet Protocol (IP). IP and MAC addresses are still required to transfer the contents, which is the destination address [7–9]. Finding and maintaining routes in IP-based routing is a stagnant challenging issue.

NDN-based VANET has numerous advantages over TCP/IP network protocols. The contents of NDN are named by themselves rather than by other communication parties, which improves efficiency and reduces routing overhead. NDN nodes are responsible for storing the contents before sending them to the requesting node, as well as storing the information in the data structure of NDN nodes, which consists of communication interfaces. This procedure can improve the network’s overall performance in terms of content delivery. TCP/IP still has some mobility concerns, and NDN offers a viable approach for addressing and resolving these issues.

To cope with the issues in IP-based VANET architecture and to tackle the advantages and benefits of NDN architecture, an NDN-based vehicular cloud and mobile IoT architecture is designed.

Vehicular Cloud Computing (VCC) has many advantages as compared with the Internet cloud. As it is well-known, many of the driver’s questions are specific to our area and have local significance. VCC provides a self-organized and efficient model of the local environment to answer these questions. The vehicles serve as a cloud for this purpose, allowing multiple services to be developed, maintained, and consumed. By boosting the storage capacity and processing capability of vehicles, VCC provides more leverage. It creates a cloud from a collection of vehicle resources via which cars can communicate efficiently to increase the capabilities of vehicles [5].

To tackle the benefits of these two networking architectures, i.e., VCC and NDN, an efficient NDN-based vehicular cloud and mobile IoT architecture is designed. To do so, first, the system model and its routing strategy for NDN-based vehicular cloud and mobile IoT architecture are introduced, which describes how vehicles share information with each other or through RSU, using cloud and mobile IoT resources. The routing strategy specifies how vehicles to cloud are formed, how cloud resources are discovered and shared, and how tasks are collected, published, and shared, among each other. Secondly, a priority-based content distribution mechanism for this architecture to improve QoS is presented.

1.1. Contributions. The following are the main contributions of this research:

(i) Using priority-based content dissemination, we design an innovative and efficient architecture for NDN-based vehicular cloud and mobile IoT to improve QoS.

(ii) We prioritize traffic flow messages into four categories: emergency, urgent, average, and least important.

(iii) The categorized messages are prioritized based on the NDDS method with constrained location and deadline.

(iv) By improving QoS requirements, multisensor data fusion of vehicular cloud and mobile IoT networks will be effective and more efficient.

(v) To simulate the proposed priority-based content dissemination in NDN-based vehicular cloud and mobile IoT architecture, we used the CloudSim toolkit, a well-known cloud computing platform.

(vi) We demonstrate that implementing the proposed technique improves the VANET’s QoS significantly.

1.2. Paper Organization. The first section contains an introduction to NDN and VANET, as well as contributions. Section 2 describes the background study and related work. Sections 3 and 4 contain the system model and simulation results. Section 5 presents the conclusion.

2. Related Work

The QoS requirements of NDN-VCC are considered while designing an effective architecture for NDN-VCC. A comprehensive literature review is performed for this purpose, encompassing VANETs, NDN background, NDN-
2.2. NDN-Based VANET System. Vehicular networks have recently drawn a lot of effort and attention from both industry and academia, resulting in a decent set of standards and protocols, as well as making it profitable. Due to the obviously increased communication in VANET, which demonstrates the content-centric technique, determinations and struggles have been made to test the feasibility of NDN, as well as the core communications design of VANET. Various research solutions have been presented to deploy NDN for VANET by enhancing core NDN and establishing new architecture or design options. Using the Interest Data Exchange Model (IDEM), many VANET communication scenarios can be displayed over NDN as a communications architecture. The vehicle serves as a content forwarder, consumer, and supplier in this architecture. RSUs and core network devices that have cache capabilities can also operate as intermediate forwarders. Several vehicles may request a single map to download or watch a video on today's IP-based network. As a result, the original producer, who can be recognized by their IP address [11], may be able to satisfy all demands. The same requests with various source IP addresses pass from each car to the network under those conditions, producing communication jamming and lowering QoS and network coordination. The high mobility of the vehicle makes communications difficult and has a negative impact on scalability. NDN-based network communications, on the other hand, rely on searching for content names and network caching utilities. Separating material from its source will vastly increase content accessibility. Rather than relying on producer availability, the needed data should be collected from any nearby vehicle. Furthermore, the NDN design straightforwardly handles mobility.

2.3. NDN-Based Vehicular Cloud Architecture. Figure 2 depicts the overall network concept of the NDN-driven VANET-based clouds that build both Mobile Vehicular Cloud (MVC) and Temporary Parked Vehicular Cloud (TPVC) to use public cloud services. When needed, a VuC vehicle (both parked and mobile) leverages public cloud services for various applications like traffic information, etc. This architecture is made up of several levels, which we will go over below.

Physical resources like vehicle nodes, data centers, and cloud architectures have traditionally been located in the architecture’s bottom tier. The in/out stages of the NDN paradigm help these physical resources do their tasks. Cloud computing services such as resource and infrastructure abstraction and virtualization are performed by the cloud manager, which can be either a vehicle or a cloud service provider, depending on the VC paradigm. Each NDN node maintains data structures such as PIT, FIB, and CS that are altered in the communication paradigm in the event of an NDN paradigm. This architecture’s organization and management component are in charge of controlling and monitoring processes including resource service discovery, management, and resource selection, among other things. As illustrated in Figure 3 and in the design scheme, certain tasks for offloading data, such as an offload manager, scheduler, and application management, as well as download results, are carried out in the top layer of the architecture [5]. Instead of IP, NDN uses a hierarchical naming convention for routing; the author proposes a method for vehicular named clouds, and the scheme’s general form is as follows.

Cloud Type/(Phase: Discover/Setu/Use/Operation)/Application/AsaService/Res/Sig/

Cloud Type: The type of cloud that the vehicle uses, such as TPVC, MVC, PC, and VuC, is identified.

Phase: it is the operation/activities of the vehicles that use any of the given cloud types. As a result, when it sends an interest signal, the phase component is employed.

App: This particular type of application will be used in the cloud.

As a service: Identifies the type of services or operation for which the interest is generated.

Res: it is the collection of resources required for a vehicle or probable to deliver a specific service, such as storage, processing content, computation, and so on.

Sign: Determine the importance of the information being processed/sends to the vehicle.

2.4. Scheduling Schemes in VANET and NDN-Based VANET. Researchers have worked on content schedules using a variety of strategies, such as the priority-based content propagation strategy described by [12], which divides content delivery into two categories: safety content and nonsafety material. Safety-related content has been prioritized over non-safety-related content. Furthermore, content
prioritization is based on the time before the material expires and the quantity of the requests. The timing of safety messages is determined by the message deadline and relative distance. A number of consumer interests and content to be shared for nonsafety content will be prioritized. They used wireless communication for all content sharing [13]. Multimedia processing is separated into four subphases in this proposed dynamic priority base architecture for multimedia cloud computing. A dedicated computing cluster calculates each subphase. For ensuring on-time delivery to a wide range of multimedia end-user’s varying priorities, computing resources are dynamically allocated to each computing server based on workload priority queues. A deadline-based routing scheme that prioritizes safety messages is proposed in [14]. The queued requests are given several types of priorities by this scheduling mechanism. If N vehicles enter the RSU zone, these vehicles are assigned a vehicle-id; and RSU receives the vehicle-id and request-id from the registered vehicle identification, and then the RSU uses the DBR algorithm to prioritize the messages. The authors offer a new RSU scheduling strategy for supplying vehicular infrastructure base data services across numerous RSUs [15]. The RSUs are hooked together. Coaction’s approach is used to transfer both safety and nonsafety data; these multiple RSUs can lower the rate of deadline failure and response time. The suggested approach aimed to reach as many automobiles as possible with information. The change in the number of vehicles, vehicles velocity, and RSU radius across multiple RSU environments allows a performance assessment in a variety of scenarios.

3. Proposed Scheme

For information storage, retrieval, computing, management, and worldwide networking, the cloud can play a key role in the VANET [16]. Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) are the three types of traditional cloud distribution models. Vehicles Using Clouds (VuCs), Vehicular Clouds (VCS), and Hybrid Vehicular Clouds (HVCs) are the three main frameworks for VANET-based clouds. The proposed priority-based scheduling strategy for NDN-based vehicular cloud and the network model for NDN-driven VANET-based clouds are discussed below.

3.1. System Model for NDN-Based Vehicular Cloud.

This section describes the system model for NDN-based vehicular cloud, of how the vehicles share their information with each other or through RSU, using cloud resources. Following is the routing strategy [17]: how the vehicles form a cloud, how the cloud resources are discovered and shared through or in between the vehicles, how the task is collected, published, and shared, etc., as shown in Figure 4.

3.1.1. Cloud Resource Discovery. To obtain the contents, a vehicle may run an application and become a cloud leader. This vehicle also recruits members, such as cars and RSUs that exist inside the current domain range. These members can then use their resources to build a vehicle cloud. This application is responsible for defining and maintaining the resource search
The search range might be defined as a certain distance, road portion, or intersection that includes significant resource types. The cloud leader sent out a broadcast message to all nodes within the search range, requesting resources. These nodes collaborate with the cloud leader to share resources.

3.1.2. Cloud Formation. When the cloud leader receives the resources, reply to the select nodes that organize in a cloud. This selection mechanism is responsible for ensuring the exact operation of the application and for performing minimizing the total resources used in the cloud.

3.1.3. Task Assignment and Result Collection. The cloud leader divides or splits the application into many tasks, which are then distributed among cloud members based on their resources’ accessibility and availability. A cloud table, which contains task assignment information and member IDs, is maintained by the cloud leader. The results are sent to the cloud leader once the assignment has been completed by cloud members.

3.1.4. Content Publishing and Sharing. The leader gathers all task results from members, processes them, and then saves or publishes the output content. If a node is unable to process the task result or store the published contents, the cloud leader oversees another vehicle cloud for data processing or content storage. The published content is disseminated over the whole network, making it available for future queries as well as searchable for new requesters. If a request is received with matching content in storage, the cloud leader and members can forward the content.

3.1.5. Cloud Maintenance. When a cloud member wants to leave the vehicle cloud, it sends the cloud leader a leaving message. The leader chooses a replacement node that responds to resource requests in the content discovery phase and has sufficient resources to finish the work left by a leaving member. The leader now delegates the work to a new cloud member and updates the cloud table.

3.1.6. Cloud Release. When the cloud leader abandons the cloud or ceases to use it, the resources are released. The resources that have been released can be used by other clouds. The leader then sends a cloud release message to all members before removing the cloud member list.

3.2. Design Priority-Based Content Dissemination Mechanism. In this paper/work an NDN-based vehicular cloud strategy for improving QoS via broadcasting.
content depending on priority is proposed. The Data Queue (DQ) and Processing Queue (PQ), as well as a scheduler, are used in this technique [12]. The PQ is made up of two buffers: the Data Buffer (DB) and the Interest Buffer (IB). The interest in vehicles is stored in IB, whereas the content is stored in DB, as can be seen in Figure 5. The scheduler is in charge of evaluating the content in the PQ and then storing it in the DQ according to the priority order to set policies for broadcasting the content such as emergency, urgent, average, and least data. According to this policy, emergency data is given the highest priority by the scheduler, followed by urgent, average, and least important data, which are saved in the PQ and sorted in DQ.

A priority scheduling algorithm based on a new way is employing NDDS to improve QoS in NDN-based vehicular clouds, as well as to estimate relative distance. Because the data name does not change in mobility, ICN supports the mobility environment, such as vehicle networks. We include the situation information by exact time and location as (for example, accident-Friday11a.m.-Location: MN-[X, Y]) in the Information Object because the Emergency message is related to human life and is usually reserved by location as well as specific time (for example, the information of an Emergency is valuable, so it measures the distance from their original location) [18].

Since the emergency contents have a short deadline, the information is double-checked to make sure it is still useful or has not expired. If a deadline is approaching, the information must be removed; otherwise, it will be transmitted to the appropriate application layer.

Emergency services are assigned the highest priority, followed by urgent, least, and average services. Because emergency services are concerned with human life, they will be bound by specific deadlines and places. As a result, in our suggested approach, each message deadline size and relative distance are all measured as parameters. In addition, as the vehicle receives safety alerts in advance, we will compute the producer velocity from the consumer, which is stated as

\[ MP = f(\Delta T, Drel, Sm, \Delta V). \]  

Here, MP stands for message priority, T stands for message deadline, Drel stands for relative distance, Sm stands for size of the message, and V is for velocity change.

\[ MP = \ln \left( \frac{(\Delta V + 1)}{(D_{req} + 1) \times (\Delta T + 1) \times (Sm)} \right) \]  

From (2), it is clear that \( \Delta T \), Drel, and Sm have an inverse relationship with message priority. Messages with a shorter
deadline and smaller size will be given higher priority to ensure that they arrive at their destination in a timely manner. Messages with a shorter distance between provider and content consumer will also be given high priority as shown in Figure 6.

3.3. Walk-Through Example. This section describes a walk-through example. In VANET vehicles are equipped with OBU which allows vehicles to communicate with each other. There are two types of messages in VANET, i.e., time-critical and non-time-critical messages. The time-critical messages consist of deadlines that are constrained by location. We call these time-critical messages emergency messages. In VANETs, all contents have a deadline, which means that these contents are processed within that specific time, and it is very important to process these contents within that specific time. For these reasons, the deadline for these contents is checked to ensure that it is a valid or outdated deadline. If the contents are valid, they are forwarded to the application layer; otherwise, the outdated contents will be discarded.

For better and fast communication in NDN-based VANET, we proposed priority-based scheduling in NDN-based vehicular cloud. The priority of the contents is assigned based on the content expiration time, i.e., deadline and the size of request, and its location distance, respectively. The messages or contents that have the smallest deadline, size, and distance will be assigned first in the scheduling queue with the help of a scheduler and so on.

Table 1 shows twelve vehicles that want to send a request for different services. For example, Car_1 wants to send a request for asking the Nearest Petrol pump, Car_2 sends a request for asking the Nearest ATM, Car_3 sends a request for helping the Rescue Helpline, and so on.
Table 2 will categorize the messages into emergency, urgent, average, and least and assign the specific values for that in order to represent the messages type as shown below.

Table 3 shows the priority-based scheduling in NDN-based vehicular cloud in which the highest priority is given to emergency and then urgent average and least. We used the NDDS method for priority-based scheduling in which the priority is given to the messages or contents according to the smallest deadline, size, and distance. The messages that have the smallest deadline and size, as well as relative distance, will be assigned first in the scheduling queue and so on as shown in Table 3.
In Table 3 the request R3 will be processed first because it has the smallest deadline and size as well as relative distance; after that R5, R10, R11, R6, R2, R1, R4, R9, R8, R12, R7 will be processed. The suggested priority basis scheduling algorithm determines the weight of each message based on its deadline, size, and relative distance. We gather the length and deadline of each message, as well as the relative distance, and then arrange them in ascending order based on the average length and deadline of each message, as well as the relative distance from the precise location. For each communication, the average difference is estimated as the relative distance from the exact location based on the deadline and size. The messages/request that has the smallest deadline and size as the relative distance from exact location will be sent first and then the other messages are processed as shown in Table 3.

### 3.4. Priority-Based Interest Packets and Data Forwarding Packets in NDN-VC

In this section, we explain the algorithm of priority-based interest packets and data forwarding packets in NDN-VC. Table 4 lists the symbols used in the proposed algorithm.

The main steps of the proposed algorithm are given in Algorithm 1 below.

The interest and data forwarding mechanism of the proposed scheme is discussed in the algorithm. When a vehicle wants to get data or acquire some information about the road situation, it forwards an interesting message to neighboring vehicles. When vehicles receive this interest, it checks the required interest data in their content store. If the interest data is not found in CS, then check PIT whether there is an entry available or not. If an entry is found, it will discard the interest; otherwise, an entry will be created in the PIT by adding the speed, size of the interest message, relative distance, and deadline of the vehicles and forwarding the interest to FIB. If the content is found in the CS, it will schedule the content and forward it according to its priority. Priority should be given to forwarding messages/content by the following procedure. First, it will categorize the contents into four categories: emergency, urgent, average, and least. If the content type is "hospital emergency, police help, control loss, natural disaster, etc." we classified these contents as "Emergency" and assigned value type $\alpha_1$, i.e., Emergency & Value $\alpha_1$. Similarly, if the content type is "Information about the nearest bank, nearby airport, etc.", we classified these contents as "Urgent" and assigned value type $\alpha_2$, i.e., "Urgent & Value $\alpha_2$". Similarly, if the content type is "Information about car parking, next traffic signal, etc.", we categorized these contents as "Average & Value Type" $\alpha_3$, i.e., "Average & Value $\alpha_3$." Similarly, if the content type is "entertainment services such as songs, videos, games, etc.", we classified these contents as "Least" and assigned value type $\alpha_4$, i.e., Least & value $\alpha_4$. Once the contents/messages are classified as an emergency, urgent, average, or least important, they are prioritized based on the minimum deadline, relative distance, and data size as min (DL, SC, DV). For each content type, we have assigned the priority values. For example, the Emergency message types have assigned the priority value $\beta_1$, the Urgent message types have assigned the priority value $\beta_2$, the Average message types have assigned the priority value $\beta_3$, and the Least message types have assigned the priority value $\beta_4$. Furthermore, these contents are prioritized based on the minimum deadline, relative distance, and data size, which is calculated as min (DL, SC, DV).

| S. No. | Messages | Data services | NDDS | Remarks |
|-------|----------|---------------|------|---------|
| 1.    | R3       | Emergency content | 3787.2 | R3 will be sent first |
| 2.    | R5       | Emergency content | 4384.6 | R5 will be sent second |
| 3.    | R10      | Emergency content | 5407.2 | R10 will be sent third |
| 4.    | R11      | Urgent content   | 5924.8 | R11 will be sent fourth |
| 5.    | R6       | Urgent content   | 6529.6 | R6 will be sent fifth |
| 6.    | R2       | Urgent content   | 6690.6 | R2 will be sent sixth |
| 7.    | R1       | Average content  | 7327.2 | R1 will be sent seventh |
| 8.    | R4       | Average content  | 7828.7 | R4 will be sent eighth |
| 9.    | R9       | Average content  | 8558.9 | R9 will be sent ninth |
| 10.   | R8       | Average content  | 8574.9 | R8 will be sent tenth |
| 11.   | R12      | Least content    | 8896.2 | R12 will be sent eleventh |
| 12.   | R7       | Least content    | 9138.8 | R7 will be sent twelfth |

### Table 4: List of abbreviations.

| Symbol | Description |
|--------|-------------|
| V      | Vehicles    |
| CN     | Contents name |
| C      | Contents    |
| CS     | Content store |
| PIT    | Pending interest table |
| SV     | Speed of vehicles |
| SC     | Size of content packets |
| DV     | Distances between vehicles |
| INT    | Interest of vehicles |
| PQ     | Processing queue |
| DQ     | Data queue |
| IB     | Interest buffer |
| DB     | Data buffer |
| DL     | Deadline |
| DP     | Data packet |
When V generates an INT (Name of Content)
(2) INT is processed in PQ
(3) PQ contains IB & DB
(4) If CN is not in PIT of IB then
(5) If C is not found in CS, then
(6) PIT add (Interest Packet)
(7) SV —— Speed (sender and receiver vehicles)
(8) SC —— Size (Content Packet)
(9) DV —— Distance (sender and receiver vehicles)
(10) DL—– Deadline (Time of Content to be delivered)
(11) Else
(12) If C found in IB
(13) Schedule C of INT from DB
(14) INT of C can be Categorized into the following C types;
(15) C Category = (Emergency, Urgent, Average, Least)
(16) If C types == (Hospital Emergency, Police Help, control loss, & natural disaster, etc.)
(17) Assign C type = Emergency & value = α1
(18) Else If C types == (Information about Nearest Bank, Nearby Airport, etc.)
(19) Assign C = Urgent & type value = α2
(20) Else If C types == (Information of car parking, next traffic signal, etc.)
(21) Assign C = Average & type value = α3
(22) Else If C types == (Infotainment services such as songs, video, games, etc.)
(23) Assign C = Least & type value = α4
(24) Scheduling the C (According to the following Assign Priority)
(25) If C type value = α1
(26) Calculate the Priority for α1 = min (DL * SC * DV)
(27) Assign Priority Values for α1 == β1
(28) End If
(29) If the Priority Value == β1
(30) Gives the Highest Emergency Priority
(31) Forward C (According to the above Assigned Priority)
(32) Else If C type value = α2
(33) Calculate the Priority for α2 = min (DL * SC * DV)
(34) Assign Priority Values for α2 == β2
(35) End If
(36) If the Priority Value == β2
(37) Gives the Urgent Priority
(38) Forward C (According to the above-Assigned Priority)
(39) Else If C type value = α3
(40) Calculate the Priority for α3 = min (DL * SC * DV)
(41) Assign Priority Values for α3 == β3
(42) End If
(43) If the Priority Value == β3
(44) Gives the Average Priority
(45) Forward C (According to the above-Assigned Priority)
(46) Else If C type value = α4
(47) Calculate the Priority for α4 = min (DL * SC * DV)
(48) Assign Priority Values for α4 == β4
(49) End If
(50) If the Priority Value == β4
(51) Gives the Least Priority
(52) Forward C (According to the above-Assigned Priority)
(53) Else If DP not received
(54) Re-Disseminate INT C in the network
(55) End If
(56) End If
(57) End If

Algorithm 1: Proposed algorithm of priority-based interest and data forwarding packets.
4. Simulation and Evaluation

For the simulation of our proposed work, a CloudSim toolkit is used for scheduling the contents on priority. The CloudSim toolkit [19, 20] is used for the system modeling and its behavior of cloud components such as Virtual Machines (VMs), data centers, and resource provisioning policies.

4.1. Simulation Setup. We create two scenarios for the simulation and evaluation of the proposed NDN-based vehicular cloud architecture. Scenario-1 consists of NDDS based priority scheduling and scenario-2 consists of nonprioritized messages as FCFS strategy. For this reason, we use the CloudSim toolkit for the simulation of our proposed NDN-based vehicular cloud architecture by adopting scheduling policies based on NDDS. So, in CloudSim we created a data center, having a processing rate of 1000 Million Instructions Per Second (MIPS) and memory of 512 MB. Table 5 shows the configuration set up for the simulated cloud and Table 6 shows configuration detail.

4.2. Scenario-1: Experimental Evaluation of NDDS Based Priority Scheduling. We assume that several requests are generated for different purposes. These requests are in the form of cloudlets in the simulation. In the first step, we got the length and deadline and distance for 20 cloudlets and then categorized the cloudlet task. Then we found the average length and deadline as well as the minimum distance of each categorized cloudlet task sorted in ascending order in the lists. Then the average difference is calculated for each cloudlet based on deadline, size, and minimum distance, and the cloudlets that have the smallest deadline, size, and minimum distance are assigned first in the scheduling queue. In this way, we will categorize the data and assign priority on the basis of deadline and size as well as to measure the relative distance. The highest priority is given to the emergency data, then urgent, average, and least data.

4.3. Simulation Result. In this section, the execution time for each task of cloudlets is calculated in the cloud by adopting the NDDS based scheduling policy. We take 20 cloudlets and assign a task to them based on NDDS. The execution time is calculated for each task of cloudlets. Table 7 shows the calculated execution time for all 20 cloudlets existing in the start and finish time through which the execution time of each cloudlet is calculated.

4.4. Scenario-2: Experimental Evaluation of FCFS Forwarding Strategy. We also created the FCFS data forwarding strategy for the proposed NDN-based vehicular cloud architecture in the CloudSim toolkit. In this, we got 20 cloudlets and performed the processing according to the FCFS strategy. Table 9 shows the experimental results in terms of execution time for FCFS. Also, Table 10 shows total calculated execution time for FCFS strategy based on the sum of start and running time and the total execution time is calculated for 05, 10, 15, and 20 cloudlets as shown in Table 10 and Figure 8.

4.5. Comparison Analysis of Scenario-1: NDDS Based Priority Scheduling vs. Scenario-2: FCFS Based Forwarding Strategy. Figure 9 and Table 11 show the comparison analysis of NDDS-based priority scheduling vs. FCFS basis forwarding strategy based on the sum of the start and running time calculated. Table 11 shows the comparison analysis of execution time for Scenario-1 vs. Scenario-2 and Figure 9 shows the comparison of NDDS based priority scheduling vs. FCFS basis forwarding strategy in terms of the computed execution time. Table 12 and Figure 9 show the comparison of execution time for NDDS based priority scheduling vs. FCFS based on the sum of start and running time and the total execution time is calculated for 05, 10, 15, and 20 cloudlets as well, which shows better results than FCFS messages.

Table 5: Basic configuration of simulated cloud.

| Cloudlet  | 20 |
|-----------|----|
| Datacenter| 1  |
| Broker    | 1  |
| Virtual machines | 2 |

Table 6: Configuration.

| System architecture | X64 |
|---------------------|-----|
| RAM (GB)            | 2   |
| Processor (GHz)     | 1.33|
| Storage (MB)        | 10000|
| MIPS                | 1000|
| Bandwidth (MBPS)    | 1000|

The execution time of five cloudlets is calculated based on the time taken by start and finish time as (Execution Time = Finish Time – Start Time), and the average execution time is calculated for five cloudlets and then 10, 15, and 20 cloudlets as shown in Table 8 and Figure 7.
### Table 7: The life cycle of Scenario-1.

| S. No. | Cloudlet ID | Message type | Data center ID | VM ID | Time  | Start time | Finish time |
|--------|-------------|--------------|----------------|-------|--------|------------|-------------|
| 1      | 1           | Emergency    | 2              | 0     | 1      | 0.1        | 1.1         |
| 2      | 12          | Emergency    | 2              | 1     | 1.11   | 0.1        | 1.21        |
| 3      | 18          | Emergency    | 2              | 0     | 1.04   | 1.1        | 2.15        |
| 4      | 2           | Emergency    | 2              | 1     | 1.31   | 1.21       | 2.53        |
| 5      | 8           | Emergency    | 2              | 0     | 1.44   | 2.15       | 3.59        |
| 6      | 5           | Urgent       | 2              | 1     | 1.59   | 3.59       | 5.17        |
| 7      | 13          | Urgent       | 2              | 0     | 1.77   | 4.1        | 5.87        |
| 8      | 7           | Urgent       | 2              | 1     | 1.87   | 5.17       | 7.05        |
| 9      | 11          | Urgent       | 2              | 0     | 2.02   | 5.87       | 7.89        |
| 10     | 0           | Urgent       | 2              | 1     | 2.04   | 7.05       | 9.08        |
| 11     | 14          | Average      | 2              | 0     | 2.19   | 7.89       | 10.08       |
| 12     | 15          | Average      | 2              | 1     | 2.19   | 9.08       | 11.27       |
| 13     | 9           | Average      | 2              | 0     | 2.55   | 11.27      | 13.82       |
| 14     | 17          | Average      | 2              | 1     | 2.67   | 12.61      | 15.27       |
| 15     | 10          | Average      | 2              | 0     | 2.75   | 13.82      | 16.57       |
| 16     | 19          | Least        | 2              | 1     | 2.85   | 15.27      | 18.12       |
| 17     | 16          | Least        | 2              | 0     | 2.91   | 16.57      | 19.48       |
| 18     | 3           | Least        | 2              | 1     | 2.99   | 18.12      | 21.12       |
| 19     | 4           | Least        | 2              | 0     | 2.04   | 7.05       | 9.08        |
| 20     | 6           | Least        | 2              | 1     | 2.19   | 7.89       | 10.08       |

| No. of cloudlets | Message type | Execution time | Start time | Finish time |
|------------------|--------------|----------------|------------|-------------|
| 05               | Emergency    | 5.9            | 4.66       | 10.58       |
| 10               | Urgent       | 8.85           | 21.26      | 29.99       |
| 15               | Average      | 11.5           | 45.37      | 56.86       |
| 20               | Least        | 14.17          | 76.39      | 90.56       |

**Figure 7:** Calculated execution time for Scenario-1.
Table 9: The life cycle of Scenario-2.

| S. No. | Cloudlet ID | Message type | Data center ID | VM ID | Time  | Start time | Finish time |
|--------|-------------|--------------|----------------|-------|--------|------------|-------------|
| 1      | 0           | Least        | 2              | 0     | 41.83  | 0.1        | 41.93       |
| 2      | 1           | Average      | 2              | 1     | 68.55  | 0.1        | 68.65       |
| 3      | 3           | Urgent       | 2              | 0     | 80.57  | 41.93      | 122.5       |
| 4      | 5           | Emergency    | 2              | 1     | 94.54  | 68.65      | 163.18      |
| 5      | 2           | Least        | 2              | 0     | 93.62  | 122.5      | 216.12      |
| 6      | 7           | Average      | 2              | 1     | 107.53 | 163.18     | 270.72      |
| 7      | 4           | Urgent       | 2              | 0     | 106.47 | 216.12     | 322.59      |
| 8      | 9           | Emergency    | 2              | 1     | 120.53 | 270.72     | 391.25      |
| 9      | 6           | Least        | 2              | 0     | 119.31 | 322.59     | 441.89      |
| 10     | 11          | Average      | 2              | 1     | 133.52 | 391.25     | 524.77      |
| 11     | 8           | Urgent       | 2              | 0     | 132.15 | 441.89     | 574.04      |
| 12     | 13          | Emergency    | 2              | 1     | 146.52 | 524.77     | 671.29      |
| 13     | 10          | Least        | 2              | 0     | 144.99 | 574.04     | 719.03      |
| 14     | 15          | Average      | 2              | 1     | 159.52 | 671.29     | 830.81      |
| 15     | 12          | Urgent       | 2              | 0     | 157.83 | 719.03     | 876.85      |
| 16     | 17          | Emergency    | 2              | 1     | 172.52 | 830.81     | 1003.33     |
| 17     | 14          | Least        | 2              | 0     | 170.67 | 876.85     | 1047.52     |
| 18     | 19          | Average      | 2              | 1     | 185.51 | 1003.33    | 1188.84     |
| 19     | 16          | Urgent       | 2              | 0     | 183.51 | 1047.52    | 1231.03     |
| 20     | 18          | Emergency    | 2              | 1     | 198.72 | 1188.84    | 1387.56     |

Table 10: Calculated execution time for Scenario-2.

| S. No. | Run time | Start time | Finish time |
|--------|----------|------------|-------------|
| 5      | 379.11   | 233.28     | 612.38      |
| 10     | 587.36   | 1363.86    | 1951.22     |
| 15     | 741.01   | 2931.02    | 3672.03     |
| 20     | 910.93   | 4947.35    | 5858.28     |
Table 11: Comparison analysis of execution time for Scenario-1 vs. Scenario-2.

| No. of cloudlets | Execution time | Start time | Finish time | No. of cloudlets | Execution time | Start time | Finish time |
|------------------|----------------|------------|-------------|------------------|----------------|------------|-------------|
| 1                | 1              | 0.1        | 1.1         | 1                | 41.83          | 0.1        | 41.93       |
| 2                | 1.11           | 0.1        | 1.21        | 2                | 68.55          | 0.1        | 68.65       |
| 3                | 1.04           | 1.1        | 2.15        | 3                | 80.37          | 41.93      | 122.5       |
| 4                | 1.31           | 1.21       | 2.53        | 4                | 94.54          | 68.65      | 163.18      |
| 5                | 1.44           | 2.15       | 3.59        | 5                | 93.62          | 122.5      | 216.12      |
| 6                | 1.58           | 2.53       | 4.01        | 6                | 107.53         | 163.18     | 270.72      |
| 7                | 1.59           | 3.59       | 5.17        | 7                | 106.47         | 216.12     | 322.59      |
| 8                | 1.77           | 4.1        | 5.87        | 8                | 120.53         | 270.72     | 391.25      |
| 9                | 1.87           | 5.17       | 7.05        | 9                | 119.31         | 322.59     | 441.89      |
| 10               | 2.02           | 5.87       | 7.89        | 10               | 133.52         | 391.25     | 524.77      |
| 11               | 2.04           | 7.05       | 9.08        | 11               | 132.15         | 441.89     | 574.04      |
| 12               | 2.19           | 7.89       | 10.08       | 12               | 146.52         | 524.77     | 671.29      |
| 13               | 2.19           | 9.08       | 11.27       | 13               | 144.99         | 574.04     | 719.03      |
| 14               | 2.53           | 10.08      | 12.61       | 14               | 159.52         | 671.29     | 830.81      |
| 15               | 2.55           | 11.27      | 13.82       | 15               | 157.83         | 719.03     | 876.85      |
| 16               | 2.67           | 12.61      | 15.27       | 16               | 172.52         | 830.81     | 1003.33     |
| 17               | 2.75           | 13.82      | 16.57       | 17               | 170.67         | 876.85     | 1047.52     |
| 18               | 2.85           | 15.27      | 18.12       | 18               | 185.51         | 1003.33    | 1188.84     |
| 19               | 2.91           | 16.57      | 19.48       | 19               | 183.51         | 1047.52    | 1231.03     |
| 20               | 2.99           | 18.12      | 21.12       | 20               | 198.72         | 1188.84    | 1387.56     |

Table 12: Comparison of calculated execution time for Scenario-1 vs. Scenario-2.

| No. of cloudlets | NDDS based priority scheduling | FCFS based forwarding strategy |
|------------------|-------------------------------|-------------------------------|
| 05               | 5.9                           | 5.9                           |
| 10               | 8.85                          | 8.85                          |
| 15               | 11.5                          | 11.5                          |
| 20               | 14.17                         | 14.17                         |
| Total execution time for 20 tasks is 40.42 | Total execution time for 20 tasks is 2618.41
5. Conclusion

The paper presents a priority-based content dissemination technique in NDN-vehicular cloud for improving quality of service (QoS). Based on the NDDS, with constrained location and deadline, traffic flow messages are prioritized into four categories: emergency, urgent, average, and least. The messages/contents with the minimum deadline, size, and distance will be assigned first and prioritized first in the scheduling queue and so on. Emergency content must be delivered as soon as possible because they are related to human life and must always be given top priority. Thus, among all messages/contents, the highest priority is given to the emergency, then urgent, average, and least messages. The priority of the contents is determined by the content expiration time, also known as the deadline, as well as the amount of the data request and its location distance. With the help of the scheduler, the messages or contents with the lowest deadline, size, and distance will be assigned first in the scheduling queue and so on. Besides, the study demonstrates using numerical results obtained with the CloudSim toolkit. The results show that the processing time of emergency messages is minimum as compared to the urgent, average, and least messages which significantly improves the NDN-based VANET's quality of service.

Data Availability

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

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This research was supported by the Researchers Supporting Project number (RSP2022R476), King Saud University, Riyadh, Saudi Arabia.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, “A survey of information-centric networking,” IEEE Communications Magazine, vol. 50, 2012.

[2] M. A. Yaqub, S. H. Ahmed, S. H. Bouk, and D. Kim, “Enabling critical content dissemination in vehicular named data networks,” in Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems, pp. 94–99, Honolulu, Hawaii, October 2018.

[3] C. Bian, T. Zhao, X. Li, and W. Yan, “Boosting named data networking for efficient packet forwarding in urban VANET scenarios,” in Proceedings of the The Twenty First IEEE International Workshop on Local and Metropolitan Area Networks, Beijing, China, April 2015.

[4] M. Amadeo, C. Campolo, and A. Molinaro, “Priority-based content delivery in the internet of vehicles through named data networking,” Journal of Sensor and Actuator Networks, vol. 5, no. 4, p. 17, 2016.

[5] R. Hussain, S. H. Bouk, N. Javaid, A. M. Khan, and J. Lee, “Realization of VANET-Based cloud services through named data networking,” IEEE Communications Magazine, vol. 56, no. 8, pp. 168–175, 2018.

[6] N. F. A. Hussein Al-Omaisy, E. A. Sundararajan, and Towards Vanet-Ndn, “A framework for an efficient data dissemination design scheme,” in Proceedings of the 2019 International Conference on Electrical Engineering and Informatics (ICEEI), Bandung, Indonesia, July 2019.

[7] S. Wang, W. Wu, and Z. Zhi, “Varied priority-based data forwarding via NDN in VANET,” in Proceedings of the 2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), pp. 574–581, Leicester, UK, August 2019.

[8] S. H. Ahmed, S. H. Bouk, D. Kim, D. B. Rawat, and H. Song, “Named data networking for software defined vehicular networks,” IEEE Communications Magazine, vol. 55, no. 8, pp. 60–66, 2017.

[9] S. Tiennoy and C. Saivichit, “Using a distributed roadside unit for the data dissemination protocol in VANET with the named data architecture,” IEEE Access, vol. 6, no. c, Article ID 32612, 2018.

[10] R. A. Rehman, T. D. Hieu, H. M. Hong-Min Bae, S. H. Sung-Hoon Mah, and B.-S. Kim, “Robust and efficient multipath Interest forwarding for NDN-based MANETs,” in Proceedings of the 2016 9th IFIP Wireless and Mobile Networking Conference (WMNC), pp. 187–192, Colmar, France, July 2016.

[11] L. Zhang, D. Estrin, and J. Burke, “Named Data Networking (NDN) Project Named Data Networking (NDN) Project,” 2010, NDN-0001.

[12] I. Khan, T. Zhang, X. Xu, S. Shan, A. Khan, and S. Ahmad, “Priority-based Content Dissemination in Content Centric Vehicular Networks,” in Proceedings of the 2018 2nd IEEE Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC), pp. 2005–2009, Imcec, Xi'an, China, May 2018.

[13] A. Ali, H. Liu, A. K. Bashir et al., “Priority-based cloud computing architecture for multimedia-enabled heterogeneous vehicular users,” Journal of Advanced Transportation, vol. 2018, Article ID 6235379, 12 pages, 2018.

[14] D. Patel, R. Patel, and N. Patel, “Cloud Assisted Adaptive Priority Scheduling using Deadline Based Reservation Strategy for VANET,” in Proceedings of the 2018 Second International Conference on Computing Methodologies and Communication (ICCMC), pp. 442–445, Iccmc, Erode, India, February 2018.

[15] K. Kyungsoon Bok, S. Seungwan Hong, J. Jongtae Lim, and J. Jaesooyoo, “A multiple RSU scheduling for V2I-based data services,” in Proceedings of the 2016 International Conference on Big Data and Smart Computing (BigComp), pp. 163–168, Hong Kong, China, January 2016.

[16] V. A. Siris, C. N. Ververidis, G. C. Polyzos, and K. P. Liolis, “Information-Centric Networking (ICN) architectures for integration of satellites into the Future Internet,” in Proceedings of the 2012 IEEE First AESS European Conference on Satellite Telecommunications (ESTEL), Rome, Italy, October 2012.

[17] E. Lee, E.-K. Lee, M. Gerla, and S. Oh, “Vehicular cloud networking: architecture and design principles,” IEEE Communications Magazine, vol. 52, no. 2, pp. 148–155, 2014.
[18] N.-T. Dinh and Y. Kim, “Information-centric dissemination protocol for safety information in vehicular ad-hoc networks,” Wireless Networks, vol. 23, no. 5, pp. 1359–1371, 2017.

[19] A. Wadhonkar and T. Deepti, “An Analysis of Priority Length and Deadline Based Task Scheduling Algorithms in Cloud Computing,” IJCSN, vol. 5, pp. 360–364, 2016.

[20] M. Adnan, J. Iqbal, A. Waheed et al., “On the design of efficient hierarchic architecture for software defined vehicular networks,” Sensors, vol. 21, no. 4, p. 1400, 2021.