CFEC: An ultra-low latency Microservices-based In-Network Computing Framework for Information-Centric IoVs

Muhammad Salah ud din 1, Muhammad Atif ur Rehman, 2, and Byung Seo Kim 2

1Hongik University
2Affiliation not available

October 31, 2023

Abstract

The advancement of vehicular onboard units (OBUs) has led to compute-intensive and delay-sensitive vehicular applications. Undeniably edge-assisted static roadside computing terminal (sRCT) offers immediate computations, a surge of smart vehicles and intensive computation requests during crowded hours may overload the sRCT, leading to performance degradation and intolerable delays. Therefore, to facilitate proximate computations and achieve ultra-low latency, this article envisions a Consortium of mobile vehicular Fog, Edge, and Cloud (CFEC) an ultra-low latency microservices-centric in-network computing framework for vehicular Named Data networks (VNDN). CFEC develops a fog-profiler-assisted mobile vehicular fog based on vehicles’ mobility patterns and available resource characteristics to ensure reliable computation offloading and reverse-path stability in a dynamic vehicular environment. Furthermore, CFEC introduces an intermediary ZTMC controller that effectively filters out underutilized sRCTs and routes computation requests to nearby, filtered sRCTs, thus minimizing transmission time and accelerating computations even during crowded hours. Simulations results revealed that CFEC significantly reduces computational satisfaction delays by up to 32.5%, 48.5%, and 31.9%, 51.025% against varying interest and node rates, respectively while in extreme traffic conditions, CFEC achieved an impressive computation satisfaction ratio of around 85% compared with benchmark schemes.
CFEC: An ultra-low latency Microservices-based In-Network Computing Framework for Information-Centric IoVs

Muhammad Salah ud din, Student Member, IEEE, Muhammad Atif ur Rehman, and Byung-Seo Kim, Senior Member, IEEE

Abstract—The advancement of vehicular onboard units (OBUs) has led to compute-intensive and delay-sensitive vehicular applications. Undeniably edge-assisted static roadside computing terminal (sRCT) offers immediate computations, a surge of smart vehicles and intensive computation requests during crowded hours may overload the sRCT, leading to performance degradation and intolerable delays. Therefore, to facilitate proximate computations and achieve ultra-low latency, this article envisions a Consortium of mobile vehicular Fog, Edge, and Cloud (CFEC) an ultra-low latency microservices-centric in-network computing framework for vehicular Named Data networks (VNDN). CFEC develops a fog-profiler-assisted mobile vehicular fog based on vehicles’ mobility patterns and available resource characteristics to ensure reliable computation offloading and reverse-path stability in a dynamic vehicular environment. Furthermore, CFEC introduces an intermediary ZTMC controller that effectively filters out underutilized sRCTs and routes computation requests to nearby, filtered sRCTs, thus minimizing transmission time and accelerating computations even during crowded hours. Simulations results revealed that CFEC significantly reduces computational satisfaction delays by up to 32.5%, 48.5%, and 31.9%, 51.025% against varying interest and node rates, respectively while in extreme traffic conditions, CFEC achieved an impressive computation satisfaction ratio of around 85% compared with benchmark schemes.

Index Terms—Named Data Networking, vehicular Networks, Edge, Fog, Cloud, Internet of Things, Microservices.

1 INTRODUCTION

The rapid technological development and advancement in computing, processing, sensing, and wireless communication technologies as well as their integration with vehicular onboard units (OBU) [1] has brought a slew of compute-intensive and delay-sensitive vehicular applications such as augmented reality (AR) based driving assistance [2], and smart 3d-navigation. These applications require high resources, low latency response, and a high communication rate to provide real-time services to the user [3]. Considering the limited locally available computational resources, a vehicle may never solely perform real-time computations within the strict latency requirements [4], [5].

To cope with the ever-increasing computation demands of consumer vehicles, cloud computing solutions are utilized, which enable consumer applications to offload their computations to the remote cloud [6]. Routing the requests towards the cloud reduces the computation overhead at the expense of high latency and bandwidth consumption [7][8] which may not be appropriate for delay-intolerant scenarios, as the long delays may cause serious consequences.

To obviate these issues, an extension of the remote cloud termed Edge computing has been introduced with the idea to install the infrastructure points such as roadside edge units (RSU Edge) [9]–[11] and push the cloud resources at the proximity of consumer. Undeniably, the RSU Edge nodes are equipped with a handsome amount of resources, these units are extremely resource-constrained compared with the remote cloud. The aforementioned schemes (e.g., [9]–[11]) only target the effective resource utilization of Edge terminals rather than utilizing the onsite vehicles resources.

Considering the urban traffic, the rapid increase in smart vehicles and their compute-intensive application requests may highly overload the RSU. The situation exacerbates in peak hours with the abrupt increases in the number of passing vehicles. Due to finite resources, an RSU mounted at a crowded location such as the intersection of the main roads may never solely fulfill the computation requests offloaded by the multitude of diverse-natured vehicular applications [6] [12]. The overloaded RSU routes the computation request towards the remote cloud at the expense of long delays. The delayed results reception may hinder the real-time decision-making of the delay in-tolerant vehicular application leading to financial, human, and infrastructural losses.

At present, the vehicular applications adopt the wireless access in vehicular environments (WAVE)[13] architecture for inter-vehicle and vehicle-to-infrastructure alerts messages without having any IP overhead. However, for multi-hop communication, the WAVE architecture utilizes TCP/UDP/IPV6 [14] to perform the packet routing.

By utilizing the unique IP address, the source vehicle locates the host vehicle to establish an end-to-end communication path to initiate communication. However, IP address
assignment and retaining the end-to-end communication path with the same IP addresses in a continuously changing environment is significantly complex due to the following reasons.

- **Infrastructure support**: IP address allocation requires infrastructural support. However, the vehicular networks fall under the category of Adhoc (self-organized) networks which do not have infrastructure support [15].

- **Unique address Identification**: As the vehicles frequently change their positions, locating a particular vehicle’s IP address is very challenging.

- **End to End path establishment**: In address-centric communication the participating vehicles require an end-to-end communication path to initiate the communication. However, retaining end-to-end connectivity in a mobile environment is pretty difficult.

To handle these limitations, information-centric networking (ICN) and its realization “Named Data Networking (NDN)” has emerged as a potential candidate [16], especially in a vehicular environment. With the name-based and content-centric communication philosophy, NDN effectively handles intermittent connectivity and high mobility issues by decoupling the services or content from any specific location.

Given the intrinsic characteristics of the vehicular environment and shared wireless medium, the conventional Interest/Data broadcast-based vanilla NDN’s content retrieval mechanism results in a broadcast storm in the network. The condition becomes adverse when the consumer applications require fast processing of a huge amount of generated data for real-time decisions making.

Several efforts (such as [15], [17]–[20]) have been devoted to the literature to investigate the effective utilization of NDN in vehicular networks to provide real-time data delivery within latency requirements. These schemes mainly focus to control the number of Interest/Data packet transmissions using various mechanisms such as hop count information, maintaining the location information of producer nodes, name centrality, and inter vehicles contact duration. However, it is significantly difficult to avoid Interest/Data flooding by retaining and utilizing these factors in a continuously evolving environment.

To address these limitations, CFEC (consortium of mobile vehicular Fog, Edge, and Cloud) presents a collaboration of mobile vehicular fog, roadside edge units, and cloud, with the precise aim to facilitate the consumer vehicle with proximate computations with minimal network and computing resource utilization. We develop a fog profiler \((f_p)\) assisted mobile vehicular fog (MVF) to provide on-site computations to delay in-tolerant vehicular applications. A via virtual weighted vehicular forwarding base \((vWVFIB)\) is developed to ensure reliable computations offloading, and reverse-path stability in a dynamic environment. Moreover, A zonal traffic monitoring and control (ZTMC) is introduced that develops a virtual reactive Forwarding Information Base \((vRfib)\) comprised of idle sRCTs to facilitate consumer applications with imminent computations during peak hours. Furthermore, A microservices-based computation offloading mechanism to effectively distribute the computation tasks across potential vehicular nodes and sRCTS while the adoption of NDN (with certain modifications) as an underlying communication protocol significantly improves the network performance via inherent in-network caching, request aggregation, and name-based forwarding features.

The core contributions CFEC are summarized as follows:

1) To ensure reliable in-network computations, CFEC develops a fog profiler \((f_p)\) that manages mobile vehicular fog (MVF) based on several heterogeneous vehicular attributes to facilitate the consumer applications with proximal computations within the strict latency requirements.

2) CFEC extends the vanilla NDN, adopts the VIKOR method, and proposes state-of-the-art virtual weighted vehicular FIB \((vWVFIB)\) to ensure reliable computations offloading, and reverse-path stability in a continuously changing vehicular environment.

3) CFEC develops an intermediary ZTMC controller (between sRCT and cloud layer) provisioned with the global view (i.e., physical location, network and compute resources) of underlying sRCT units. We formulate an efficient strategy that filters the underutilized sRCT units and develop an innovative virtual reactive sRCT FIB (VR-FIB) that enables the ZTMC to offload the incoming computation request toward the optimal sRCT via VR-FIB aiming to facilitate time-sensitive and compute-intensive consumer applications within the maximum tolerable latency during peak hours.

4) CFEC devises a microservices-based naming schema and computation offloading mechanism to ensure judicious resource utilization, optimize bandwidth consumption and provide fast computations to delay-sensitive and compute-intensive vehicular applications.

The rest of this paper is organized as follows: Section 2 provides some brief background on NDN and some related work. Section 3 provides an overview of monolithic and microservices architecture. The proposed CFEC system is presented in Section 4. Section 5 is devoted to the performance evaluation and finally, Section 6 concludes the paper.

## 2 Background & Related Work

### 2.1 Vehicular Named Data Networking: An overview

The NDN-based content retrieval process in vehicular networks triggers when a consumer vehicle interested in fetching some named-based content broadcasts an Interest packet in its coverage radius. The receiving vehicle in the transmission radius of the consumer performs the Interest and Data processing procedures on intrinsic NDN data structures such as pending Interest table (PIT), content store (CS), and forwarding information base (FIB). Upon Interest packet reception, the consumer first enquires about the existence of the same name Interest packet entry in the PIT. If the PIT entry is found the interest aggregation process will be performed otherwise the CS check will be executed. If CS check is successful, the node responds with a Data packet back in the reverse path on the incoming interface else a new PIT entry is created and the Interest packet is forwarded toward the upstream node via FIB lookup. The Data packets follow the chain of PIT entries in the reverse path to reach back to the consumer vehicle. If no PIT entry
is found against an incoming Data packet, the Data packet is considered unsolicited and is discarded.

2.2 Related Work: NDN/ICN based Vehicular Communication

The NDN is regarded as a promising future Internet architecture due to its inherent content-centric communication philosophy. The enchanting features such as Named-based content acquisition, name base Interest/Data forwarding, and in-network caching enable NDN to meet the challenging vehicular networking environment. In this regard, several efforts have been devoted to the effective integration of NDN with vehicular networks aiming to resolve the broadcast storm problem and achieve efficient and on-time content delivery. Table 1 presents a comparison of various state-of-the-art NDN-based schemes developed for vehicular networks based on several metrics such as broadcast storm mitigation, Reverse path maintenance, collaborative offloading (based on Fog, Edge, and cloud) and in-network computations whereas, a comprehensive review of the current state-of-the-art Vehicular Named Data Networking schemes is provided below.

A scheme named NameCent [15] was developed to address the issue of data broadcast storm in Vehicular Named Data Networks (VNDN). Namecent proposed a forwarding strategy based on Name centrality and received signal strength (RSSI). The name centrality is computed based on the number of copies of the same interest received by a particular vehicle while the RSSI indicates the link stability. Both these metrics are utilized in optimal next hop forwarder selection. However, this approach faces challenges in retaining centrality values in a dynamic environment and does not consider the coalition time between vehicles, leading to frequent packet retransmissions and redundant broadcasts. This can result in congestion and resource overutilization, limiting the effectiveness of the scheme.

The authors in [17] proposed "Cost-Efficient Data Retrieval Based on Integration of VC and NDN" for cost-efficient data retrieval in vehicular networks. CDRVC generates a vehicular cloud (VC) where each member shares and caches copies of the data, increasing the chances of data retrieval in the network. A stable vehicular backbone comprised of forwarding vehicles is also developed to achieve unicast communication and avoid redundant transmissions. However, in sparse networks where no backbone node is present, the node may need to broadcast the interest to fetch the desired content, which can lead to increased network traffic and reduced efficiency.

Another well-known scheme named CODIE also addressed the broadcast storm problem was developed in [18]. CODIE introduced the hop-count field in the Interest Packet and the data dissemination limit (DDL) in Data packets. The hop-count field is used to limit the number of intermediate nodes that forward the Interest packet, while the DDL field ensures that the packet is not forwarded beyond the actual consumer. This approach effectively reduces network resource utilization and improves network performance in vehicular NDNs. CODIE may face performance issues in dense networks where multiple nodes may take part in interest and Data packets transmissions resulting in network resource overutilization and QoS degradation.

In [23] authors targeted to enhance the QoS of different vehicular communication paradigms, including V2V, V2I, and V2R. Their approach focused on enhancing popularity-based content caching by inspecting the performance of cache placement and the replacement of vanilla NDN to increase the efficiency and effectiveness of content delivery in vehicular networks. The authors in [24] proposed a search and routing technique to retrieve requested content from the closest router’s cache in vehicular networks. While the proposed work alleviates retrieval latency to some extent, the frequent probing may turn into an additional burden on the network resulting in network congestion and collisions.

A robust forwarder selection mechanism was proposed in [25]. The proposed work developed the Neighbors Satisfied list (NSL) and Recent Satisfied List. Periodic exchange of RSL is required to keep the NSL up to date which results generate significant communication overhead. To enhance the packet delivery ratio authors proposed a Density Aware Delay Tolerant interest forwarding scheme for NDN-based vehicular networks [20]. In this work, each vehicle maintains neighbor information, and a rebroadcast defer timer is developed to mitigate the broadcast storm in the network.

Several state-of-the-art location-based ICN architectures have been proposed such as LOICEN [19], and LOCOS [22]. LOICEN rely on opportunistically acquiring the location information of vehicles to find potential content providers with the desired content in their locally established cache. LOICEN assume that there is a high possibility of availability of the same type of content in the cache of already discovered content providers as well as their neighboring vehicles. To manage provider information, LOCOS [22] includes the provider’s location, timestamp, and content prefix with the FIB table. On receiving an Interest packet, both schemes select the next-hop forwarder that is closer to the provider’s location. If a potential forwarder is unavailable, both schemes use the conventional NDN forwarding mechanism to locate the provider.
3 Monolithic and Microservice Architecture:

A monolithic architecture is a conventional model where the entire system is developed as a single indivisible unit with tightly coupled services. Migrating a particular module of a system between servers requires the migration of the complete monolith, resulting in bandwidth consumption, long delays, and network congestion [12]. Scarce bandwidth and computation resources in vehicular environments may cause performance issues for compute-intensive real-time delay-intolerant applications using monolithic architecture, resulting in delays, high latency, and network congestion.

To overcome the challenges, microservices (MS) architecture has emerged as a potential candidate due to its potentialities such as less communication and computation overhead, optimal resource utilization, low bandwidth utilization, and low maintenance cost [12]. Compared with the monolithic architecture, the MS architecture decomposes the application into several atomic MS able to execute independently. MS architecture enables independent deployment of MS over compute terminals without affecting other MS in the same application. Each MS performs a specific task, requires fewer resources, and allows efficient migration with minimal bandwidth consumption.

4 CFEC: Proposed Work

Before delving into the operation of CFEC, we first shed light on CFEC system architecture, as well as the system model and assumptions.

4.1 CFEC System Architecture

The CFEC system architecture (depicted in Fig. 1) provides a resourceful platform for compute-intensive and latency-strict vehicular applications. The proposed architecture is comprised of four linked layers by the power and importance from the bottom to the top such as:

- **Autonomous mobile vehicular fog (MVF):** Since modern autonomous vehicles are equipped with handsome computing, storage, and communication resources. CFEC exploits the available resources of multiple on-site vehicles and transforms them into gigantic on-site computing machinery referred to as autonomous mobile vehicular fog (MVF) which is administered by the fog profiler \( f_p \) that manages the MVF in terms of fog management, resource orchestration, and microservices-based task allocation to the MVF members, MVF member failure management, and load balancing. The \( f_p \) avoids requests offloading towards the sRCT and distributes the computations among the most reliable MVF members which are not only closer to the \( f_p \) but also to the requesting consumer vehicle through Virtual Weighted Vehicular FIB (vWVFIB, details in 4.4). Due to their closeness to the consumer vehicle, they provide proximal computations with reduced transmission and processing time compared with the Edge and cloud offloading.

- **Static Rendezvous Computing Terminals (sRCTs):** This layer is composed of lightweight Edge computing nodes...
co-located with the fixed roadside unit (RSU). CEFC calls these computing terminals Static Rendezvous Computing Terminals (sRCT). When the $f_p$ fails to locally facilitate the consumer computation request it offloads the computational request toward sRCT.

- **Zonal Traffic Monitoring Center (ZTMC):** CEFC safely assumes that modern cities are comprised of multiple service zones [26] where every zone has its zonal traffic Monitoring center (ZTMC). ZTMC manages the traffic by coordinating the sRCTs, CCTV cameras, and other infrastructure units ensuring road users have a comfortable and safe journey.

In CEFC, the overutilized sRCTs, instead of offloading the computing task directly towards the remote cloud forward the request to ZTMC to perform efficient computations. The ZTMC has a global view of all the sRCTs such as physical location, workload condition, available resource information, etc. During intense traffic conditions, the ZTMC avoids cloud offloading, exploits the underutilized sRCTs in its vicinity, and offloads the computing tasks towards them to avoid the long latency and backhaul bandwidth utilization that may occur in the case of CC offloading.

- **Centralized Cloud (CC):** CC corresponds to several diverse-natured server clusters equipped with powerful computation, communication, and storage facilities. ZTMC offloads the computation request towards the CC when fails to perform the computation task locally or via underutilized sRCTs.

**4.2 System Model and Assumptions**
CFEC considers a hybrid vehicular environment built upon 1) MVF comprised of several mobile vehicles with heterogeneous OBUs capabilities, 2) the fixed sRCTs, ZTMC, and CC 3) microservices-based application requests with heterogeneous computation and latency requirements.

To further elucidate the CEFC underlying working mechanism, consider a hybrid vehicular environment composed of $L$ vehicles (i.e., mobile fog nodes) denoted as $F = \{f_1, f_2, f_3, \ldots f_L\}$. These fog nodes organized themselves into MVF headed by the $f_p$. In addition, we consider a set of $K$ computing terminals i.e., sRCTs denoted by $C = \{c_1, c_2, c_3, \ldots c_k, \ldots c_K\}$, one ZTMC per city zone, and a $CC$. Moreover, CEFC considers $N$ vehicular applications denoted by $A = \{a_1, a_2, a_3, \ldots a_n, a_N\}$, $M$ microservices represented by $M=\{m_1, m_2, m_3, \ldots m_m, m_M\}$. Each application i.e., $a_n \in A$ is composed of a collection of microservices such as $a_n = \{a_{n_1}, a_{n_2}, a_{n_3}, \ldots a_{n_k}\}$.

CEFC assumes that each vehicle is equipped with a location module i.e., GPS for location estimation (e.g., polar coordinates, cartesian coordinates), speed sensors, and direction sensors in addition to computation, storage, and communication units [1]. Furthermore, the sRCT communicates with the neighbors via wireless backhauls while the wired connection is employed for the communication between sRCT and ZTMC, and ZTMC to CC.

Table 2 presents the notations and their definitions used in the CEFC.

**4.3 CEFC: Operation**
A generalized view of the CEFC’s entire operation can be visualized in Fig. 2, while the detailed description and functionality of each building block are provided as follows.

**4.3.1 $f_p$ administered MVF development and resource orchestration:**
The $f_p$ administered MVF development and Resource orchestration procedure involves the following stages.
TABLE 2: Notations and definitions

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
<th>Notation</th>
<th>Definition</th>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFEC</td>
<td>Consortium of Fog Edge Cloud</td>
<td>MVF</td>
<td>Mobile vehicular fog</td>
<td>NDN</td>
<td>Named Data Networking</td>
</tr>
<tr>
<td>OBU</td>
<td>Onboard unit</td>
<td>CC</td>
<td>Centralized cloud</td>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>sRCT</td>
<td>Static roadside computing terminal</td>
<td>f_p</td>
<td>Fog profiler</td>
<td>WAVE</td>
<td>wireless access in vehicular environement</td>
</tr>
<tr>
<td>ZTMC</td>
<td>Zonal traffic monitoring and control</td>
<td>f_s</td>
<td>Fog member vehicle</td>
<td>PFT</td>
<td>Pending Interest Table</td>
</tr>
<tr>
<td>C_t</td>
<td>Coalition time</td>
<td>VIKOR</td>
<td>Vlekriterijumsko</td>
<td>FIB</td>
<td>Forwarding information base</td>
</tr>
<tr>
<td>MCDM)</td>
<td>multicriteria decision-making</td>
<td>vWVFIB</td>
<td>Virtual Weighted Vehicular FIB</td>
<td>CS</td>
<td>Content Store</td>
</tr>
<tr>
<td>VR-FIB</td>
<td>Virtual Reactive sRCT FIB</td>
<td>VNDN</td>
<td>Vehicular Named Data Networks</td>
<td>MS</td>
<td>Microservice</td>
</tr>
</tbody>
</table>

Fig. 3: MVF-discovery namespace.

Fig. 4: MVF coalition namespace.

1) **MVF development and f_p recruitment**: CFEC considers a wide range of smart vehicles present on the roads during the rush hours, exploits their available resources, and transforms them into MVF (headed by f_p) with a considerable amount of resources, that offers on-site computations to the strict latency and resource-hungry applications. The f_p recruitment procedure may incorporate various metrics such as remaining distance to reach the destination, current location, bandwidth, vehicle speed, computation resources, and battery power. However, “the specific procedure for recruiting f_p” is out of the scope of our work.

2) **MVF Association**: A vehicular node performs the following steps in order to join and participate in MVF operations. A detailed mechanism is provided as follows.

   a) **MVF discovery**: Initially, a smart mobile vehicle intending to join the MVF inquires about the presence of MVF in its vicinity by forwarding an MVF-discovery Interest packet as shown in Fig. 3 where:

      i) f_p Info: indicates that the consumer vehicle wants to inquire about the presence of available f_p in its vicinity.
      ii) vehId: This component denotes the unique vehicular id (i.e., number) Interested to join and participate in MVF operations.
      iii) vehSpeed: The vehicle’s current speed.
      iv) currentLoc: Current location coordinates of the vehicle.
      v) destLoc: Final destination coordinates of the vehicle.
      vi) direction: Corresponds to the movement direction of the vehicle.

   b) **Consumer to MVF coalition and resource sharing**: A consumer vehicle may receive multiple MVF-discovery Data packets (shown in Fig. 3) from available f_p in its coverage radius. However, it selects and associates with f_p, which has a low relative velocity difference and the maximum proportion of overlapping routes toward the final destination. The rationale for is to ensure reliable computations, durable connectivity, and MVF stability in a highly mobile vehicular environment. To do so, the consumer vehicle forwards the MVF coalition interest as shown in Fig. 4.

      i) f_pcoalition: indicates that the underlying vehicle wants to associate with the MVF.
      ii) f_pId: corresponds to the unique identifier of the f_p, to whom the consumer vehicle wants to associate.
      iii) vehId: represents the unique vehicle number.
      iv) compRes: The available computing that the vehicle can offer for computations.
      v) bw: The bandwidth information of the vehicle.
      vi) <MSInfo>: The list of microservices-based application operations, a vehicle can perform.

   c) **Consumer coalition approval**: Upon receiving the coalition Interest, the f_p computes the coalition time (C_t) of the consumer vehicle, executes the Virtual Weighted Vehicular FIB (vWVFIB) procedure (explained in subsection 4.4), and acknowledges the consumer’s membership by forwarding the MVF-coalition Data packet (shown in Fig 4) carrying the C_t between f_p and consumer vehicle in the Data payload.

   The motive for maintaining the C_t is to keep the member vehicle aware of its coalition period with the f_p, and to avoid computation offloading towards the f_p after the expiration of C_t. The mechanism effectively reduces computation losses and network resource over-utilization.

3) **MVF Member Failure/departure handling**: An MVF member may leave the MVF due to any unforeseeable reason such as vehicle internal fault, traffic Jams, sudden route change, etc. In order to avoid the computation losses and uphold the MVF integrity, the departing vehicle informs the f_p by forwarding the depart Interest packet as shown in Fig. 5.

   a) f_pdepart: indicates that a fog member is about to depart the MVF.
   b) f_pId: corresponds to the unique identifier of the f_p.
   c) vehId: It represents the departing member’s unique
vehicle number.

The \( f_p \) acknowledges the departing vehicle by forwarding a depart Data packet depicted in Fig. 5.

### 4.4 Virtual Weighted Vehicular FIB (vWVFIB):

As discussed previously, the MVF may be composed of a wide variety of smart vehicles bearing heterogeneous computational, and communication resources in addition to vehicular dynamics such as average speed, final destination, and direction. In our proposed \( f_p \)-administered MVF, \( f_p \) first computes the \( C_t \) between \( f_p \) and \( f_l \) to ensure the reliability of MVF. For instance, a \( f_l \) having a handsome amount of computational resources but low \( C_t \) time may leave MVF earlier. Therefore, offloading the computations toward the vehicle bearing low \( f_l \) may never provide reliable computations and result in computation losses. In addition, a vehicle bearing high \( f_l \) while low computational, and communication resources also questions the reliable computations as the consumer vehicle application may never receive the on-time computation results.

To address both limitations, CFEC extends the vanilla NDN, adopts the VIKOR (Vlekriterijumsko KOMpromisno Rangiranje, a Serbian term for “multi-criteria optimization and compromise solution”) method, and proposes state-of-the-art vWVFIB where the \( f_p \) considers both \( C_t \) and available vehicular resources in the interface ranking procedure. During the computation offloading procedure, \( f_p \) offload the request to the high-ranked interface via vWVFIB which further reduces the chances of reverse path partitioning and reliable computation results handover to the consumer vehicle.

A complete step-by-step description of each stage of the vWVFIB development process is as follows.

#### 4.4.1 Coalition time computation procedure:

Coalition time (\( C_t \)) refers to the amount of time in which the \( f_l \) remains in contact with \( f_p \). Since vehicles in vehicular networks change their position frequently depending on their speed and direction. At a particular instant, a vehicle might be in connection with another but might not at the other instant. Therefore, the vehicles’ \( C_t \) plays a pivotal role in ensuring: 1) reliable inter-MVF computation offloading 2) reverse-path stability during the Data Transfer, and 3) redundant transmissions reduction and bandwidth optimization. In addition, the available computational resources of MVF members are indispensable in providing the requested computation within the tight latency requirements.

Consider \( f_p \) with the position coordinate \((x_{fp}, y_{fp})\) moving with a certain velocity \((v_{fp})\) and direction \((\theta_{fp})\) receives a coalition Interest from a vehicle \((f_l \in F)\) having position \((x_{fl}, y_{fl})\), velocity \((v_{fl})\), and direction \((\theta_{fl})\). Since the vehicles are traveling in a particular direction in a 2-d plane [28] with a certain speed and direction, the distance between \( f_p \) and \( f_l \) can be computed using the polar coordinates \((r,\theta)\) [29], as shown in Eq.1.

\[
d_{fp+fl} = \left[ (r_{fp} \cos \theta_{fp} - r_{fl} \cos \theta_{fl})^2 + (r_{fp} \sin \theta_{fp} - r_{fl} \sin \theta_{fl})^2 \right]^\frac{1}{2}
\]

(1)

It is to be noted that, the origin point is chosen as the \( f_p \), and the radius \( r \) represents the radial distance from \( f_p \) to the \( f_l \) vehicle. The angle \( \theta \) is taken with respect to the orientation of the \( f_p \), representing the relative angular position of the \( f_l \) vehicle.

Expanding and simplifying the Eq.1

\[
d_{fp+fl} = \left[ r_{fp}^2 (\cos^2 \theta_{fp} + \sin^2 \theta_{fp}) + r_{fl}^2 (\cos^2 \theta_{fl} + \sin^2 \theta_{fl}) - 2r_{fp}r_{fl} (\cos \theta_{fp} \cos \theta_{fl} + \sin \theta_{fp} \sin \theta_{fl}) \right]^\frac{1}{2}
\]

(2)

Since \( \cos^2 \theta + \sin^2 \theta = 1 \), and \( \cos \theta_{fp} \cos \theta_{fl} + \sin \theta_{fp} \sin \theta_{fl} = \cos(\theta_{fp} - \theta_{fl}) \), by incorporating these values and simplifying Eq.2, we get \( d_{fp+fl} \) as shown in Eq.3

\[
d_{fp+fl} = \left[ r_{fp}^2 + r_{fl}^2 - 2r_{fp}r_{fl} \cos(\theta_{fp} - \theta_{fl}) \right]^\frac{1}{2}
\]

(3)

The proposed scheme employs the same procedure to compute the distance between \( f_p \) and the final destination of \( f_l \) i.e., \( d_{fp+dest_{fl}} \). The \( d_{fp+dest_{fl}} \) enables the \( f_p \) to compute the overlapping proportion of the journey covered by both \( f_l \) and \( f_p \) to reach the destination. The mechanism ensures the Fog stability and promotes reliable computation offloading and also minimizes the frequent MVF maintenance procedure.

Based on the above equation, we can compute the \( C_t \) as follows.

1) If \( f_l \) is moving ahead of \( f_p \), the \( f_l \) leaves the vicinity of \( f_p \) earlier as it has already covered some distance, therefore, \( C_t \) can be computed as follows.

\[
C_t = \frac{d_{fp+dest_{fl}} - d_{fp+fl}}{v_{fp+fl}}
\]

(4)

2) If \( f_p \) is moving ahead of \( f_l \), the \( f_p \) leaves the vicinity of \( f_l \) earlier as it has already covered some distance, therefore, \( C_t \) can be computed as follows.

\[
C_t = \frac{d_{fp+dest_{fl}}}{v_{fp+fl}}
\]

(5)

Where \( v_{fp+fl} \) is the velocity difference between \( f_l \) and \( f_p \).

#### 4.4.2 Optimal vehicular Face computation:

The \( f_p \) adopts a VIKOR method to quantify each corresponding vehicular face. VIKOR belongs to the class of multicriteria decision-making (MCDM), used for achieving the best possible decision-making in multi-factor decision problems. It is an efficient decision-making mechanism that offers a flexible technique able to handle a wide variety of heterogeneous decision variables and guide the decision-making entity to make the optimal offloading decision in continuously evolving environments such as vehicular networks.
The $f_p$ considers several heterogeneous vehicular attributes such as $C_t$ (i.e., $C_1$), computational resources ($C_2$), available BW ($C_3$), relative velocity ($C_4$) in the vWVFIB development process and organizes these attributes in a tabular form. After enlisting the values of the attributes, it is essential to extract the optimal solution for beneficial ($C_+$) and non-beneficial ($C_-$) resource value $j$, i.e., $1 < j < n$ by employing Eq.6 and Eq.7.

$$X_j^+ = [\max_i (x_{i,j}), \text{if } j \in C_+; \min_i (x_{i,j}), \text{if } j \in C_-]$$

(6)

$$X_j^- = [\min_i (x_{i,j}), \text{if } j \in C_+; \max_i (x_{i,j}), \text{if } j \in C_-]$$

(7)

Where $X_j^+$ and $X_j^-$ denote the ideal solution for beneficial and non-beneficial attributes. In case of a beneficial attribute, the highest attribute value (e.g., storage, computation, and bandwidth) is desired while the lowest attribute value (e.g., relative velocity, latency) is regarded as the best attribute value for a non-beneficial attribute. After computing the ideal solution, a weighted normalization is performed to bring the incongruous resource values to a common scale as follows.

$$u_{i,j} = \frac{w_j \times X_j^++x_{(i,j)}}{X_j^+ - X_j^-}; \text{where } i = 1..m; j = 1..n$$

(8)

Where $w_j$ is the relative importance of each factor assigned by the decision maker [30] or can be allocated by adopting the mechanism provided in [31]. After computing the weights we have to compute the group utility measure ($S_i$) that corresponds to the overall gain and the Regret factor ($R_i$) which shows the overall deviation.

$$S_i = \sum_{j=1}^{n} (u_{i,j}); \text{ where } i = 1..m$$

(9)

$$S^* = \min_i (S_i)$$

$$S^- = \max_i (S_i)$$

(10)

$$R_i = (\max_j (u_{i,j})) = \max_j \left( w_j \times \frac{X_j^++x_{(i,j)}}{X_j^+ - X_j^-} \right)$$

(11)

$$R^* = \min_i (R_i)$$

$$R^- = \max_i (R_i)$$

(12)

Finally, $f_p$ computes the priority factor $Q_i$ of each member vehicle in the MVF based on the aforementioned computations by utilizing the following equations

$$Q_i = \frac{S_i - S^*}{S^- - S^*} + (1 - \text{d}) \times \left( \frac{R_i - R^*}{R^- - R^*} \right)$$

(13)

The $\text{d}$ value determines the rule that has to be applied [30] for the final decision making e.g., The $\text{d} = 0.5$ represents the majority voting rule that prefers the group utility measure. The $\text{d} = 0.5$ corresponds to the consensus rule which ensures a balance between the group and individual group members. $\text{d} < 0.5$ prefers the vehicular node that dominates the $Q$. CFEC utilizes the consensus rule [30] as it resides between the veto and majority voting rule. The $f_p$ organizes the vWVFIB entries in the increasing order of the $Q_i$. During computation offloading, the $f_p$ offloads the computations to the vehicle that has a minimum value of $Q$ [32]. It is worth noticing that the vWVFIB is updated 1) Upon each request offloading towards the MVF member to avoid resource overutilization of a member. Meaning that the offloading towards a single $f_p$ may result in the resource bottleneck which may result in computation losses, delays, and network resource consumption. 2) Upon the expiration of $C_t$ to uphold the fog stability. This is due to the fact that the member vehicle leaves the MVF upon the expiration of $C_t$. Therefore computation requests offloading towards already left members may result in un-tolerable delays, computation losses, and bandwidth consumption.

4.5 Virtual Reactive sRCT FIB (VR-FIB)

As already discussed, ZTMC has global view of the MS-based computations offered by each sRCT, the current workload conditions, and the available as well as the total resources of sRCTs mounted in the region. It is safe to assume that each sRCT undergoes a non-identical workload during peak and off-peak hours depending on their location (e.g., crowded or non-crowded). The ZTMC aims to provide real-time computation results and relieve the overutilized sRCT by offloading a proportion of the workload toward the idle sRCT during peak hours. To do so, ZTMC devised an efficient formulation to evaluate the “health status” of each sRCT based on current workload, computational capability, storage, and bandwidth. Based on the formulation results a virtual reactive site FIB (VR-FIB) is developed where each interface is organized according to its current health status (shown in the top right corner of Fig.6).

The sRCTs health status evaluation and VR-FIB development mechanism are provided as follows.

**Problem Formulation:** An sRCT (e.g., sRCT$^t$) is considered an optimal offloading candidate if it has enough available resources to accomplish the incoming computations requests i.e.,

$$a_o \Gamma_{k}^{sRCT^t} > \text{req}^{MS_m} \forall k \in \mathbb{R}, m \in M$$

(14)

Where $a_o \Gamma_{k}^{sRCT^t}$ represent the available $k^{th}$ resources of $i^{th}$ sRCT, $\text{req}^{MS_m}$ denote the required $k$ resources to execute $m^{th}$ microservice request and $\mathbb{R}$ represents the set of resources.

In order to evaluate the overall health status of each sRCT, the ZTMC computes the individual score ($\Psi_k$) of each available resource as shown in Eq.15 and normalizes the computed individual score to a common range (i.e., $[u \ v]$), $u=0$ and $v=1$) as depicted Eq.16.

$$\Psi_k = \frac{a_o \Gamma_{k}^{sRCT^t}}{\text{req}^{MS_m}}$$

(15)

$$\Psi'_k = u + (\Psi_k) - \min (\Psi_k) \times (v - u)$$

(16)

The final health status of each sRCT is computed as follows.

$$\xi_{i}^{sRCT} = \sum_{k \in \mathbb{R}} \phi^k \Psi'_k$$

(17)

Where $\xi_{i}^{sRCT}$ corresponds the health status of $i^{th}$ sRCT node and $\phi^k$ is the decision maker’s assigned application specific weight factor of $k^{th}$ resource [31] and $\mathbb{R}$ is the set of resources.

An sRCT that has the highest health value is considered a potential offloading candidate. The ZTMC selects the optimal sRCT and offloads the requested computations via VR-FIB.
4.6 CFEC: Microservices Centric Operations

4.6.1 Microservices-based Application Scenario and Naming Schema

In the microservices architecture, an application can be partitioned into several dependent and independent microservices. The former can be offloaded and executed simultaneously while the latter may require input from other microservices. CFEC considers smart driving assistance application composed of several independently deployable and executable microservices such as:

1) Data preprocessing (MS1-DPP)
2) Object detection (MS2-OD)
3) Object representation (MS3-OREP)
4) Object recognition (MS4-OREG)

Each microservices differs from others in terms of computational requirement, latency, and QoS constraints. CFEC developed an MS-centric naming schema that allows consumer vehicles to delegate the computation requests to fp and receive the corresponding responses. This schema allows efficient and scalable processing of MS-based tasks, relieving the computational burden on consumer vehicles and improving overall performance. For instance, the following name, “CFEC:/DriverAssistance$MS1-DPP1P1P2” where the DriverAssistance represents the application name, the “$” indicate that the Interest includes a microservice computation request. The microservice name (MS1-DPP) and its input parameters (P1, P2) are separated by the vertical pipe “|” while each parameter is further separated by the “,”.

4.6.2 Microservices-based Computation Offloading and Results Handover:

Consider a smart vehicle’s dash camera captures visual content (e.g., video, or static image) required to perform specific microservices-based computations as shown in Fig. 6. The detailed task offloading and computations delivery procedure is provided as follows.

1) Member Vehicle to fp Offloading: Consider MVF-1 in Fig. 6, where the consumer vehicle “A” forward an Interest packet (MS1-I_A) towards the fp. On receiving the request, the fp first authenticates the received Interest and verifies whether adequate resources are available for computations. As shown in the figure, the corresponding fp has enough resources and available microservice code, the fp performs the local computations and transfers the Data packet i.e., MS1-D_A to the consumer vehicle.

Consider the case when the required resources or microservice code is not available (vehicle C in MVF-2 sent MS3-I_C shown in Fig. 6), the fp on interest reception consults the locally maintained vWVFIB and offloads the received MS3_I_C to the potential compute vehicle (i.e., B) bearing the lowest Q. Based on the received Interest packet, vehicle “B” identifies that the fp is demanding computations. The vehicle B executes the microservice code on received input parameters and handover the results Data packet MS3-D_C to the fp, and fp further forwards the results back to the consumer vehicle C. The scenario can be visualized in Fig. 6 (MVF-2, steps 1-4 in

Fig. 6: CFEC computations offloading and results delivery procedure
green color).

2) \( f_p \) to sRCT and inter sRCT Computations Offloading: When an \( f_p \) fails to satisfy the computation requests (due to resource unavailability, small \( C_t \) with the \( f_p \), etc., ) it offloads the task towards the sRCT mounted in the transmission radius. If the sRCT has adequate computing resources as well as microservice code, it performs the computations and handover the results to the \( f_p \) directly on the Adhoc interface (Fig. 6, MVF-2, steps 1-4 in red color).

The connection between the sRCT and the \( f_p \) may never last for a longer duration due to high mobility. For instance, the high-speed \( f_p \) may leave the vicinity of the current sRCT soon after forwarding the computation request (e.g., \( f_p \) managing MVF-3 in Fig. 6). In this scenario, request offloading is required to avoid computation losses and uphold the application quality of decision-making. However, In many vehicular applications, the task input data is much larger than the processed results. The transmission of input file between the sRCTs utilizes a high bandwidth and affect the wireless backhaul transmission efficiency.

In order to enhance the transmission efficiency with minimum bandwidth consumption, in the proposed work, the corresponding sRCT performs the computations and handover the computed results to the neighboring sRCT (i.e., mounted in the direction of \( f_p \)) instead of employing the NDN’s inherent reverse-path Data delivery mechanism. To do so, the sRCT appends a handover tag with the Data packet. The reason is to inform the neighboring sRCT to deliver Data to the \( f_p \) via wireless adhoc Interface without marking the incoming Data packet as unsolicited due to the non-availability of a corresponding PTP entry in the PIT table. The scenario of \( f_p \) to sRCT and inter sRCT results Offloading can be visualized in Fig. 6 (MVF-3, 1-5 in blue color).

3) sRCT to ZTMC Computations Offloading: The CFEC sRCT to ZTMC offloading mechanism enables the exhausted sRCT to offload application requests to the ZTMC controller instead of CC aiming to facilitate the consumer computations within tolerable latency with minimum quality loss, less transmission cost, and service latency. The sRCT appends the coverage duration (CD i.e., the time until the \( f_p \) remains in the range of sRCT) and the direction of \( f_p \) with the Interest packet. The ZTMC upon receiving the request offload towards the idle sRCT via VR-FIB.

Upon receiving the computed results the ZTMC handovers the computed results to \( f_p \):

a) In reverse path: If the compute nodes perform the computations and forward the results to ZTMC within the CD, the ZTMC forward the results in a reverse path towards sRCT, and the sRCT hand over the results directly to the consumer.

b) Via the sRCT in the direction of MVF: If upon receiving the computed results the CD expires, the ZTMC appends “handover” tag with the Data packet and forwards it to the sRCT mounted in direction of MVF. The receiving sRCT forwards the Data packet to \( f_p \) which further handover to the consumer.

The scenario of sRCT to ZTMC Computations Offloading can be visualized in Fig. 6 (MVF-4, 1-7 in brown color).

5 Performance evaluation

This section deals with detailed performance evaluation.

5.1 Simulation Setup

To demonstrate the effectiveness and potential benefits of CFEC, extensive simulations are performed in the ndnSIM—an ns-3-based network simulator. The open street maps [33], and the SUMO [34] mobility generator were integrated with the ndnSIM to envision a realistic urban vehicular traffic environment. We compared the proposed scheme with the state-of-the-art schemes named cost-efficient data retrieval based on the integration of VC and NDN (CDRVC) [17] and NameCent: name centrality-based data Broadcast mitigation in Vehicular named data networks (NameCent) [15] to check the reliability of fp-managed CFEC in terms of interest satisfaction delays (ISD), total packets processed in the network, communication overhead, and interest satisfaction ratio (ISR).

An extensive comparison of the proposed work with Edge only and Cloud only schemes is also provided to analyze the computational performance in terms of average computation satisfaction time (ACST), and backhaul traffic overhead (BTO).

In the Simulations, the proposed work adopted the geographical location of the Itaewon district, South Korea, downloaded from open street maps. We consider 5 MVFs composed of multiple vehicles and an \( f_p \) managing each corresponding MVF. We deploy 3 sRCTs and a ZTMC where the former assists the overloaded \( f_p \) and the latter avoids the cloud offloading aiming to accomplish the consumer request within the quality constraints. In the simulations, the network size is varied from 50 to 100 nodes. In addition, we consider 4 different types of microservices each of which requires different computational resources for its operation. To mimic microservices computational behavior, CFEC adopted the ndnCSIM [35] codebase which provides the fundamentals of microservices execution and nodes’ resource management. The complete simulation parameters can be visualized in Table 3.

To evaluate the performance of CFEC against the benchmark schemes, we considered the following reliability and computational performance metrics in the simulation study.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulator</td>
<td>NS-3 (ndnSIM)</td>
</tr>
<tr>
<td>Communication stack</td>
<td>NDN</td>
</tr>
<tr>
<td>Mobility generator</td>
<td>SUMO</td>
</tr>
<tr>
<td>Number of sRCTs</td>
<td>3</td>
</tr>
<tr>
<td>Wireless interface</td>
<td>IEEE 802.11p</td>
</tr>
<tr>
<td>Network size (i.e., number of vehicles)</td>
<td>50-100</td>
</tr>
<tr>
<td>Vehicle Transmission Range</td>
<td>250m</td>
</tr>
<tr>
<td>Average vehicle speed</td>
<td>50Kmph, 70Kmph</td>
</tr>
<tr>
<td>MS request rate</td>
<td>2-25 requests/sec</td>
</tr>
<tr>
<td>MS request distribution</td>
<td>Random</td>
</tr>
<tr>
<td>Simulation time</td>
<td>400s</td>
</tr>
</tbody>
</table>
• Reliability Performance Evaluation: We evaluated CFEC against the benchmark schemes by taking the following reliability metrics into consideration.

1) Interest Satisfaction delay: Interest Satisfaction delay (ISD) is defined as the time taken by the consumer’s (C) Interest to reach the potential provider node (P) (i.e., $t_{C,P}$), P’s Interest processing time (i.e., $t_{P}^{process}$), and the time taken by the computed results packet to reach back to C (i.e., $t_{P,C}$).

$$\text{ISD} = t_{C,P} + t_{P}^{process} + t_{P,C}$$

2) Total Packet processed: Packets processed refer to the total number of Interest and Data packets processed in the network.

3) Communication Overhead: The communication overhead denotes the total bandwidth consumed in the Interest and Data packet transmissions.

4) Interest Satisfaction Ratio: Interest satisfaction ratio (ISR) corresponds to the total number of Data packets ($D_{pkt}^i$) received against the total number of Interest packets generated in the network ($I_{pkt}^i$).

$$\text{ISR} = \frac{\sum_{i=1}^{N} D_{pkt}^i}{\sum_{i=1}^{N} I_{pkt}^i}$$

• Computational Performance Analysis: We analyzed the performance of CFEC including the sRCT, ZTMC, and cloud in addition to the $f_p$. To do so, we considered the following metrics.

1) Average computation satisfaction time: The average Computation satisfaction time (ACST) denotes the proportion of time an MS-centric computation request takes from its origination to the delivery of computation results.

2) Backhaul traffic overhead: The backhaul traffic overhead (BTO) is defined as the percentage of computation requests arrived at the cloud station to the total number of requests generated by the consumer node.

5.2 Evaluation Results

5.2.1 Reliability Performance Evaluation:

1) Interest Satisfaction delay (ISD): The comparison of Interest satisfaction delays (ISD) in terms of Interest rate and network size can be visualized in Fig. 7(a) and Fig. 7(b) respectively. The results show that CFEC achieved an average of 32.5% and 48.5% lower delays compared to CDRVC and NAMECENT when varying the Interest rate, respectively. Similarly, when varying the number of nodes, CFEC achieved an average of 31.9% and 51.025% lower delays compared to CDRVC and NAMECENT, respectively. The rationale is that CFEC exploits the coalition time, the available computational, and BW resources offered by the corresponding member. The former reduces frequent path re-establishment and avoids redundant packet transmissions while the latter accelerates interest processing time, reducing the overall ISD.

In contrast, the CDRVC considers only coalition time between the nodes in its offloading process, however as the number of vehicles and request rate increases, offloading requests while ignoring the resource availability highly burdened the node which in turn increases the processing delays due to resource bottleneck. Whereas, the NAMCENT only considers the name centrality in its offloading mechanism. In a highly dynamic environment, the lack of availability of nodes with high name centrality results in redundant packet broadcast and congestion in the network which as result increases the ISD.

2) Total Packets processed in the network: The total Interest and Data packets processed as a function of request rate and network size can be visualized in Fig. 8(a) and Fig. 8(b). The results show that the CFEC - $f_p$ significantly reduces the total number of packets compared with the NAMCENT and CDRVC respectively (approximately less than 40% and 70% compared with NAMCENT and CDRVC respectively). The reason is CFEC’s confined packet transmission strategy, which eliminates blind broadcasts. Meaning that the consumer vehicle instead of broadcasting the interest packet throughout the whole network, offloads the Interest packet solely to its managing $f_p$. The $f_p$ then fulfills the request itself or via the most reliable fog members, thereby significantly reducing the number of packet transmissions.

In contrast, both NAMCENT and CDRVC produced high communication overhead. In the NAMCENT, the content
producer broadcasts the packet, and the receiving nodes with a weight value higher than a specified threshold rebroadcast until the packet reaches the consumer. This process involves a large number of redundant packet transmissions. In CDRVC, when the producer node fails to find the relay with a high contact duration, it broadcasts the packet to establish a vehicular backbone to reach the host node, leading to excessive packet transmissions in the network.

3) **Communication Overhead**: The communication overhead is directly proportional to the number of packet transmissions in the network. For the analysis, we vary the number of Interest packets per vehicle per second and analyzed the overhead generated by CFEC and the benchmark schemes as shown in Fig. 9. It is clear from the results that CFEC significantly reduced the overhead compared with both schemes. The rationale is the CFEC-fp-assisted MVF generation and restricted intra MVF request offloading avoid the redundant packet transmissions (shown in Fig. 8) which in turn reduces the overall communication overhead. In contrast, the benchmark schemes involve several packet re-transmissions due to the unavailability of the potential relay node, resulting in high communication overhead.

4) **Interest Satisfaction Ratio (ISR)**: The interest satisfaction ratio (ISR) of the proposed work and the benchmark schemes as a function of Interest frequency can be visualized in Fig. 10. For ISR we vary the interest frequency from 2 interest/sec to 14 interests/sec. The result in the figure demonstrates that CFEC-fp outperforms both schemes in ISR. The reason is that CFEC developed fp-assisted MVF directly fetches the requested content from potential fog members closer to the consumer vehicle via vWVFIB. In the extreme case, the fp instead of blind broadcast requests the sRCT /or intermediary ZTMC to retrieve the content and handover to the consumer which significantly increases the ISR.

In contrast, both the NAMCENET and CDRVC blindly broadcast the interest packet in absence of a potential provider node which leads to network congestion and collisions and as a result reduces ISR.

5.2.2 **Computational Performance Analysis:**

1) **Average computation satisfaction time (ACST)**: The average computation satisfaction time (ACST) as a function of MS-centric computation requests per second can be visualized in Fig. 11. As shown in the figure, the increases in requests rate results in increased ACST. However, the CFEC significantly reduced (around 80%) the computation satisfaction time compared with the benchmark schemes. The reason is that the CFEC-developed MVF and ZTMC-managed sRCT enable proximate computations thereby enabling the consumer to retrieve the com-
putation results within miniaturized delays. In addition, the CFEC computed coalition time among the vehicles also plays a significant role in reliable reverse path development and on-time data delivery to the consumer application which consequently decreases the ACST.

![Fig. 11: Average computation satisfaction time as a function of Interest frequency](image1)

The Edge-only solutions perform well compared with the cloud-only solutions as shown in Fig.11. However, with the increase in computation request rate, the ACST increases at Edge. The rationale is that the lack of available computational resources hinders the capability of onsite computations in Edge-only solutions which compel the underlying Edge node to share its burden with the Cloud resulting in high computation satisfaction time. The Cloud-only solutions show the highest ACST compared with both CFEC and Edge-only schemes. The rationale is that all the consumer application requests directly arrive at the cloud station resulting in high BTO.

![Fig. 12: Backhaul traffic overhead](image2)

2) **Backhaul Traffic Overhead (BTO):** We measured the backhaul traffic overhead (BTO) generated by the CFEC, Edge-only, and Cloud-only schemes as shown in Fig.12. The comparison results depicted in the figure demonstrate that with the increase in computation requests, the proposed scheme generated low BTO overhead and outperformed the benchmark schemes. The rationale is that CFEC’s multi-layered computation philosophy restricts the computation requests to the MVF-managed MVF and the ZTMC-managed sRCTs utilizing \( f_p \) administered vWVFIB and VR-FIB respectively. Therefore, CFEC satisfies the major proportion of computation requests in close proximity to consumers and minimizes the backhaul traffic. In contrast, the Edge-only schemes also provide proximate computations to a certain extent, however, due to resource constraints these schemes offload the computations towards the cloud to avoid computation losses which as a result increases the BTO. The cloud-only schemes show the highest BTO compared with CFEC and Edge-only schemes. The rationale is that all the consumer application requests directly arrive at the cloud station resulting in high BTO.

6 **Conclusion**

The paper presents CFEC: a multi-layered in-network computing framework that utilizes available vehicular resources and transforms resources into MVF managed by an fp proximate computations for real-time latency-sensitive vehicular applications. CFEC’s developed vWVFIB ensures reliable computations offloading, and reverse-path stability in dynamic vehicular environments. Whereas, the intermediary ZTMC controller having a global view of sRCTs under its administration avoids cloud offloading via VR-FIB further reducing the distant transmissions, communication/computation cost, and backhaul traffic overhead. Simulation results revealed that CFEC significantly reduces transmission latency up to 32.5%, 48.5%, and 31.9%, 51.025% under varying interest and node rates, respectively. In extreme traffic conditions, CFEC achieves an impressive 85% computation satisfaction ratio and limits backhaul traffic to approximately 15% compared to benchmark schemes.

In future work, we plan to develop the improved CFEC (iCFEC) by incorporating reinforcement learning approaches to devise intelligent \( f_p \) recruitment, compute node selection mechanisms, and computation offloading strategies in order to enhance the performance of compute-intensive real-time vehicular applications. Moreover, we are also developing an optimization scheme that considers unforeseeable conditions (such as vehicle internal faults, traffic jams, sudden route changes, etc.) for more realistic and effective decision-making in iCFEC.

**ACKNOWLEDGMENTS**

This work was supported in part by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2022R1A2C1003549) and in part by the Ministry of Trade, Industry and Energy(MOTIE) and Korea Institute for Advancement of Technology(KIAT) through the National Innovation Cluster R&D program(R&D, P0015131).

**REFERENCES**


Muhammad Salah ud din (Student Member, IEEE) received his M.S degree in Computer Science from University, Islamabad, Pakistan in 2016. He is currently pursuing a Ph.D. degree in Computer Engineering from Broadband Convergence Networks Lab, Hongik University, South Korea. His major interests are in the field of wireless sensor networks (WSNs/UWSNs), Named data networking, NDN enabled Vehicular Edge/Fog computing, Metaverse and the IoTs.

Muhammad Atif Ur Rehman is a Lecturer (Assistant Professor) in the Department of Computing & Mathematics at Manchester Metropolitan University, the UK since May 2022. He received a PhD degree in Electronics and Computer Engineering from Hongik University, South Korea in Feb 2022. His research interests are in the broader areas of edge cloud computing, intelligent communication protocol design, and Metaverse.

Byung-So Kim (M’02–SM’17) received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Florida, in 2001 and 2004, respectively. From 2005 to 2007, he was with Motorola Inc., Schaumburg, IL, USA, as a Senior Software Engineer in networks and enterprises. He is a Professor at the Department of Software and Communications Engineering, at Hongik University, South Korea.