

Mobility-based Multi-layered Caching and Data Distribution in Vehicular Fog Computing

K M Nafiul Hassan

Submitted in partial fulfillment
of the requirements for the degree of

Master of Science

Department of Computer Science
Faculty of Mathematics and Science
Brock University
St. Catharines, Ontario

©*K M Nafiul Hassan*, 2023

Abstract

Intelligent transportation systems provide valuable services to drivers and passengers, whereas vehicular networks enable fundamental underlying support through communication and data sharing. While edge and fog computing offers the benefits of providing support to access data closer to devices and applications, they come at the cost of requiring ongoing communication and computation. Moreover, the significant latency of the communication makes it prohibitive to fetch data from other sources. In the context discussed, caching data is a feasible option as it helps to increase performance and make data more readily accessible. Also, finding the desired data in the cache enables services besides improving performance. This study proposes a multi-layer caching mechanism that magnifies data availability while enhancing communication between the layers. The cache in the layers is distributive and updated on time based on the demand and criteria of the requests. We also distribute data using vehicular mobility by placing limited but significant data into the cache of the vehicle. These communication and data exchange types are standardized through policies in the proposed methodology. This cache management design is extensively analyzed using established frameworks and vehicular networks through simulated environments and visual constructions. Our simulation results indicate that the proposed method improves the performance of data availability and latency in vehicular fog computing. This approach can be applied to the diverse vehicle-to-everything use cases of the intelligent transportation system.

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Professor Dr. Robson De Grande, for his guidance, encouragement, and support throughout the course of this thesis. I am deeply indebted to him for his invaluable insights, constructive criticism, and unwavering support, which have been instrumental in the successful completion of this research.

I would also like to extend my thanks to my family and friends for their love and support during the course of this research. Without their encouragement, this thesis would not have been possible. Foremost, I would like to praise and thank the almighty, who has granted countless blessing, knowledge and strength, so that I have been able to accomplish this work.

Finally, my heartfelt gratitude to Brock University and each one of them associated with this famous institution, whose enormous contributions led granting me with scholarships and research fellowships that allowed my dream come true to obtain higher studies in abroad.

Thank you all.

Contents

1	Introduction	1
1.1	Thesis Statement	2
1.2	Motivation	2
1.3	Objectives	3
1.4	Contribution	3
1.5	Structure Overview	4
2	Background	5
2.1	Vehicular Ad Hoc Networks	5
2.1.1	Vehicle-to-Vehicle (V2V)	6
2.1.2	Vehicle-to-Infrastructure (V2I)	6
2.1.3	Hybrid architecture (V2X)	7
2.2	Internet of Vehicles	7
2.3	Vehicular Cloud Computing	8
2.4	Vehicular Edge Computing	9
2.5	Vehicular Fog Computing	9
2.6	Information-Centric Networking	11
2.7	Vehicular Caching	11
3	Related Works	13
3.1	Delay aware caching	13
3.2	Edge Caching	14
3.3	Cooperative Service Caching	14
3.4	Fog Caching	15
3.5	Caching in 5g and emergence of AI	15
3.6	Remarks	16
4	Problem Statement	17

5	Hierarchical Transient Caching	20
5.1	Multi-Level Caching	20
5.1.1	Primary Layer	20
5.1.2	Fog Layer	21
5.1.3	Edge Layer	22
5.2	Cache Flow, Update, and Management	22
5.3	Distribution	24
5.4	Policies	25
6	Popularity-based Distribution	27
6.1	Fog Zone Oriented Popularity Caching	27
6.1.1	Local and Global Popular Contents	27
6.1.2	Zone Formation	29
6.1.3	TLRU Factor Function	30
6.1.4	Popularity Threshold Periods	32
6.2	Inter Fog Popular Content Distribution	33
6.3	Cache Load and Distribution	35
7	Performance Analysis	36
7.1	Simulation Setup	36
7.2	Evaluation of Multi-Level Caching	37
7.2.1	Simulation Scenario	38
7.2.2	Simulation Parameters	38
7.2.3	Performance Metrics	40
7.2.4	Protocols	40
7.2.5	Results	41
7.3	Evaluation of Popular Content Caching	44
7.3.1	Simulation Scenario	45
7.3.2	Simulation Parameters	45
7.3.3	Performance Metrics	46
7.3.4	Protocols	47
7.3.5	Results	47
7.4	Remarks	51
8	Conclusion	53
8.1	Summary	53
8.2	Future Work Directions	54

List of Tables

3.1	Summary of the related works.	16
5.1	Undertaken policy parameters.	26
6.1	Fog Nodes Role Allocation.	30
6.2	Transient cache structure.	33
7.1	Simulation parameter settings.	39
7.2	Simulation parameter settings.	46

List of Figures

2.1	Illustration of vehicle-to-vehicle communication.	6
2.2	Illustration of vehicle-to-infrastructure communication.	7
2.3	Illustration of hybrid communication architecture.	7
2.4	Example of different types of IoV communication.	8
2.5	Evolution of vehicular cloud computing (VCC).	9
2.6	General vehicular edge computing (VEC) Architecture.	10
4.1	Scenario of fog regions in an urban center.	18
5.1	Abstract view of the architecture of our approach.	21
6.1	Zone formation with fog nodes (Red dots).	29
6.2	Inter fog top cached content distribution.	34
7.1	Map of simulated scenario: Washington DC, USA.	38
7.2	Cache Hits (%) vs Vehicular Density (8 Fog controllers).	42
7.3	Latency (ms) vs Vehicular Density (8 Fog controllers).	42
7.4	Cache Hits (%) vs Fog Density (40 vehicles).	43
7.5	Latency (ms) vs Fog Density (40 vehicles).	43
7.6	Fog node position for simulation: Washington DC, USA.	44
7.7	High density mobile simulation with 200 vehicles.	45
7.8	Cache Hits (%) vs Content Popularity (α) ($CC = 5\%$).	48
7.9	Latency (ms) vs Content Popularity (α) ($CC = 5\%$).	48
7.10	Content Hop Ratio (%) vs Content Popularity (α) ($CC = 5\%$).	49
7.11	Cache Hits (%) vs Cache Capacity (CC) ($\alpha = 0.6$).	50
7.12	Latency (ms) vs Cache Capacity (CC) ($\alpha = 0.6$).	50
7.13	Content Hop Ratio (%) vs Cache Capacity (CC) ($\alpha = 0.6$).	51

Chapter 1

Introduction

A vehicular network is a form of ad-hoc network that connects vehicles and infrastructure using wireless communication. It allows for the exchange of information between vehicles and between vehicles and the infrastructure to improve traffic safety and efficiency. In recent years, there have been significant technological advancements in vehicular ad hoc networks (VANETs) that are helping to make this potential a reality. These advancements include the deployment of 5G wireless networks, the use of dedicated short-range communications (DSRC) and Cellular-V2X (C-V2X) technologies, the implementation of multi-access edge computing (MEC), and the integration of cloud computing in VANETs. These new technologies enabled the development of advanced V2V and V2I applications such as high-definition video streaming, real-time traffic updates, and augmented reality that can improve traffic safety and efficiency. With these advancements, VANETs are becoming more reliable, secure, and cost-effective, making it a well-versed, interesting research area [18].

The evaluation of vehicular cloud computing (VCC) mainly comes from VANETs. Vehicular cloud systems are large-scale network that has many challenges and prospects. The system deals with high traffic request load and communication connectivity between clouds and vehicles. Caching is part and parcel of serving high traffic at an optimal time for a large-scale system. Several studies and research have been conducted to improve the performance of caching in a vehicular cloud system [22]. Delay-aware caching, edge caching, and cooperative caching are some of the established strategies for a better caching performance [17, 38, 25]. Meanwhile, such a dynamic network where wireless connections may be unreliable or unavailable has led us to edge and fog computing strategies to process computation closer to the mobile nodes. The fogs are capable of computational handling; therefore, later on, the caches are placed in the fogs, closer to the content requester, which leads to much-improved performance

in terms of caching.

1.1 Thesis Statement

Vehicular fog computing is essential for data access to devices and applications. Caching data is a well-acclaimed strategy that enables faster responses and makes data more available. This work, thus, proposes This work proposes a hierarchical caching mechanism that increases data availability and readiness in the context of fog computing. The work also explores data distribution with vehicular mobility. Furthermore, this study can identify popular local content and its usefulness in placing it in the cache. While popular contents have been stored in the cache, the cache is always kept alive and refreshed through multi-layer fog communications. Thus, the vehicles are used to distribute the contents using their mobility from one fog region to another. These communication and data exchange types are standardized through specific policies in the system, which target higher cache hits and reduced latency. This cache management design is analyzed to suit both small and large-scale urban distribution in real-time simulated scenarios.

1.2 Motivation

Vehicular networks are fundamental to enabling connected vehicles and effectively establishing an intelligent transportation system (ITS). Reliable vehicular communication is critical for content sharing and data dissemination. Communication reliability is constantly challenged in vehicular networks due to the high mobility [13]. The high complexity of novel vehicular systems, services, and applications imposes ever-growing demands of continuous communication. More efficient and flexible coordination of resources and data has notably grown with cloud-based resource management paradigms, such as vehicular fog computing (VFC). The fundamental feature of vehicular edge and fog computing enables access to data almost instantaneously, allowing it to perform computation closer to terminal devices and applications. The shorter distance in fog computing and its structure allows SDN-based approaches to solving many of the issues inherent in the vehicular environment to implement more reliable data communication through lower latency and data loss [5, 36]. However, even with VFC, the costs involved in accessing data from sources still limit delivery to high latencies. Caching is critical in faster responses and making data available, enabling efficient services and applications to vehicles and devices. The more data

cache hit happens, the system response happens to be much faster. Still, it is a great challenge to ensure data is available in the cache considering the adversity of high vehicular mobility. Several works have attempted to adjust caching in vehicular environments, such as merging it in VFCs, designing cooperative caching, minimizing delays, and converging it to support 5G and AI [20, 25, 17, 19, 26, 35, 29]. High mobility is still an aggravating factor. Even though such previous works enhance data access and sharing, the high mobility compromises the proper distribution of cached data.

1.3 Objectives

Our goal is to provide distributed caching support to the vehicles from their closest proximity, which provides faster access to data in a limited time, generates more cache hits at a local point, and eases the dependency to route the request to the primary information center. Furthermore, the popularity-based caching in our system is destined to guide important and most frequently requested content to be shared among the fog zones so that it can provide instant responses and fetch the content at the closest point, even before the request happens. In this regard, the following sub-objectives are aimed to be achieved in this study:

- To properly distribute content among vehicle nodes considering the high mobility of vehicles and network topology changes that happen dynamically.
- To enable efficient cache management following the architectural structuring of vehicular fogs.
- To reduce hops and latency to make caches timely refreshed and readily available at any fog node and edge node.
- To map the popularity of the content and its usefulness for fulfilling near-future data requests, minimizing data fetching latency, bandwidth consumption, and control overhead.

1.4 Contribution

The contribution of this thesis highly consists of an efficient hierarchical caching strategy, which enables multiple tiers of cache nodes available to serve the requesting

nodes. This work thus has generated the following contributions in the field of cache management in VFC:

- We designed a comprehensive multi-layer caching system that incorporates structural features, such as data centers, fog controllers, and edge nodes.
- We developed a transient caching approach that allows vehicles to carry and share content among urban geographical regions. This multi-layer design with transient caching have been accepted in ACM/IEEE DS-RT 2022 [15].
- We organized the cached contents into fog zone-based popular indexing to reduce sharing random contents into the cache and make popular contents take the majority of the caching space.
- We devised cache offloading to other fog nodes based on traffic intensity, satisfying content availability in local proximity.
- We have performed extensive analyses through realistic simulations where we observed the efficiency of our system under different fog configurations, vehicular density, content distributions, and caching capacity.

1.5 Structure Overview

The remainder of this thesis is organized as follows. Chapter 2 describes the background of the study of this work. Chapter 3 lists and describes the previous works relevant to cache management in vehicular environments. Chapter 4 represents the problem statement of our thesis. Chapter 5 introduces the proposed hierarchical caching strategy with transient cache distribution. Chapter 6 introduces the work based on fog zone adhered popularity oriented caching indexing and distribution. Chapter 7 presents performance analyses and discusses the results. Finally, Chapter 8 concludes the thesis and provides future research directions.

Chapter 2

Background

Vehicular networks are emerging interests with the significant improvement of communication technology and increasing demand for data for various applications in a vehicular environment. VANETs primary goal is to make transportation safer and more efficient. This network propagates data in a wireless communication channel and a large-scale environment. Nowadays, vehicles are equipped with applications that exchange information to assist in safe transit, traffic efficiency, and infotainment purposes [2]. This vehicular application requires a massive amount of data to fulfill its purpose. Vehicles' high mobility and extensive network scale bring dynamic challenges to make data propagate fast and efficiently. Vehicle requesting data, searching data, and responding is time-consuming. Therefore, data caching comes in handy and is necessary for vehicular networks. Caching in vehicular networks gives a significant advantage to data availability and fast access to data for demanding vehicular applications. Caching proves to be very necessary for large-scale vehicular communication in the modern era. Therefore, cache strategies and management policies are being extensively explored for greater data dissemination and availability efficiency.

2.1 Vehicular Ad Hoc Networks

VANETs is a term that enables wireless communication between vehicles and other devices or roadside units. In establishing an intelligent transportation system, it is necessary to have VANETs to be effective. These networks are made up of vehicles with processing capacity and wireless communication, pedestrians, and highways, sending and receiving information from other vehicles. The network consists of nodes such as automobiles, trucks, and buses, with wireless communication interfaces and equipment attached to the roads. Vehicles communicate with each other and other

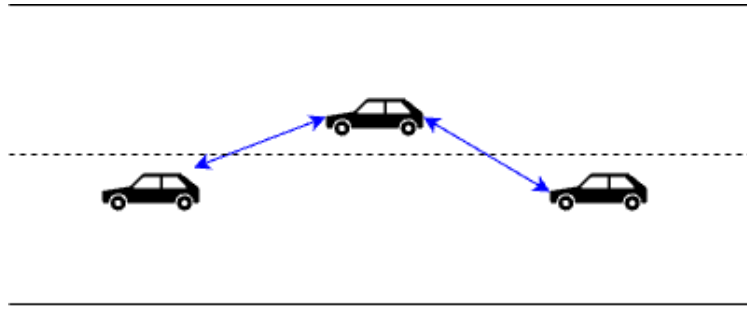


Figure 2.1: Illustration of vehicle-to-vehicle communication.

roadside infrastructures through short ranged radio frequencies, cellular networks such as long-term evolution (LTE), and 5G new radio (5G NR). Advancements in telecommunication networks allow using many architectures for vehicular networks in urban, rural, and highway environments to support existing applications. The primary goal of a VANET architecture is to allow communication between nearby vehicles and between vehicles and fixed devices on the road, leading to three possible scenarios described in the following items [18].

2.1.1 Vehicle-to-Vehicle (V2V)

This architecture enables the communication of vehicles with any other vehicles in the network. This communication technique allows data to distribute quickly and effectively in certain areas. The primary applications for this type of communication are traffic controlling, monitoring conditions of roads, and obtaining necessary information about traffic congestion to help drivers make routing decisions easily. These types of applications are more suited to urban premises. In this methodology, the data transmission time should be quick, and packet loss has to be as minimum as possible. Due to dynamic speed and high mobility of the vehicles, this communication throws challenges to transmit data faster and with bare minimum loss [18]. Figure 2.1 illustrates a general concept of V2V communication.

2.1.2 Vehicle-to-Infrastructure (V2I)

This architecture mainly allows communication vehicles with other fixed infrastructures or roadside units (RSU) in the network. Some vehicular applications require internet connections through an access point on roadsides, such as cell towers or Wi-Fi towers. Connecting to the internet enables the services from data centers, clouds, and various sources to the vehicle. For such communication and management of the

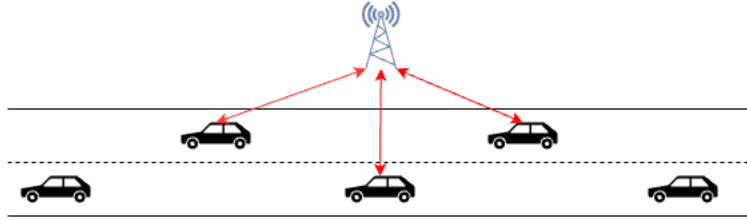


Figure 2.2: Illustration of vehicle-to-infrastructure communication.

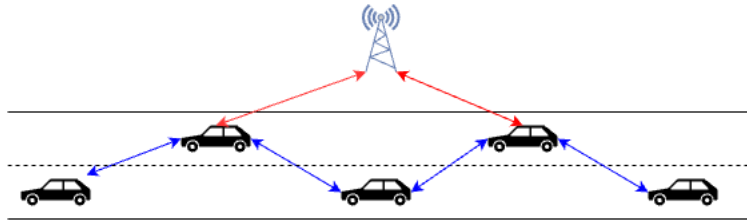


Figure 2.3: Illustration of hybrid communication architecture.

mobility of the vehicles, service exchange from one access point to another may be unnoticed by the user. The handoff techniques to be established in such communication must deal with the data flow, and streams of data over the network [18]. Figure 2.2 illustrates V2V communication.

2.1.3 Hybrid architecture (V2X)

This architecture combines both V2V and V2I architectures. A vehicle is able to communicate with the infrastructures in single hop or multiple hops. It also depends on location in relation to the point of connection with the infrastructure for different purposes [18]. Dedicated short-range communications (DSRC) are designed as wireless access in vehicular environments (WAVE) to support V2X communications, focused explicitly on enabling vehicular safety applications [8]. DSRC devices are equipped with 802.11p chip and mainly include onboard units (OBU), roadside units (RSU), and mobile devices carried by pedestrians, which enables diverse communication to the vehicles [14]. Figure 2.3 illustrates V2X communication.

2.2 Internet of Vehicles

In terms of scalability and infrastructure internet of vehicles (IoV) is an advancement of traditional vehicular ad-hoc networks (VANETs). VANET mainly refers to sending or receiving data intelligently between vehicles or other infrastructures. Many intelligent devices can collect data from the surroundings and share them over the

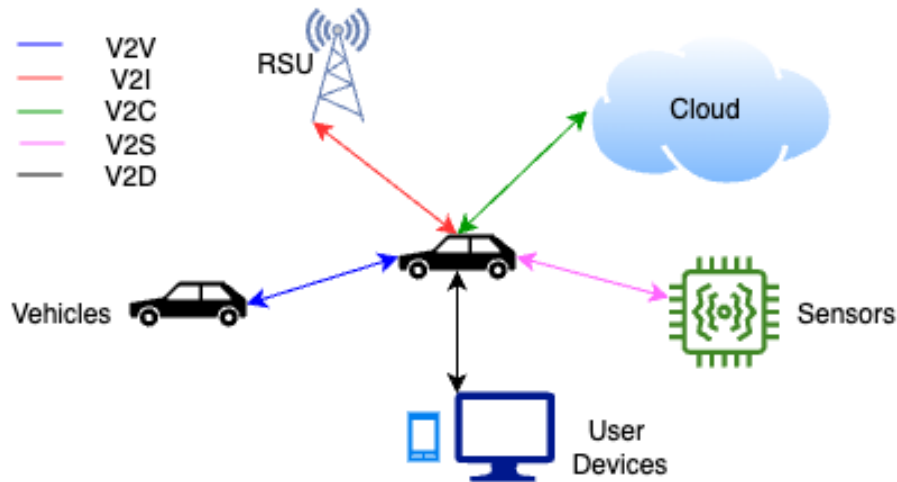


Figure 2.4: Example of different types of IoV communication.

internet, referred to as the internet of things (IoT). IoV is conceptualized by the combination of VANET and IoT. Various types of communication can be possible such as vehicle to pedestrian, vehicle to clouds, and vehicle to sensors. [10]. IoV refers to collecting data on vehicles, roads, infrastructures, and surroundings from a vehicular environment and sharing data through the internet for supporting intelligent traffic control, intelligent vehicle control, and dynamic information service in intelligent transportation systems (ITS) [11]. IoV also enables information sharing faster on communication channels and allows users to transmit valuable data to the requester and other applications efficiently and easily. Modern cloud computing (CC) architecture integration to vehicle networks gave IoV an extra dimension. These technologies enable applications, data, and resources to be saved in remote areas and servers and can be accessed whenever requested. The CC platform handles a massive quantity of data produced by the interlinked vehicles [9]. Figure 2.4 illustrates different types of communications possible through IoV.

2.3 Vehicular Cloud Computing

Vehicular cloud computing (VCC) has become very popular as a service and as a support to the applications that clouds can provide. The cloud stems and its application capabilities provide reliable services, fast computation, and dynamic resource allocation to the network [18]. VCC evolving from VANETs allows authorized users to utilize higher computational services at a low cost and capable of implementing an improved intelligent transportation system to be data shareable and available al-

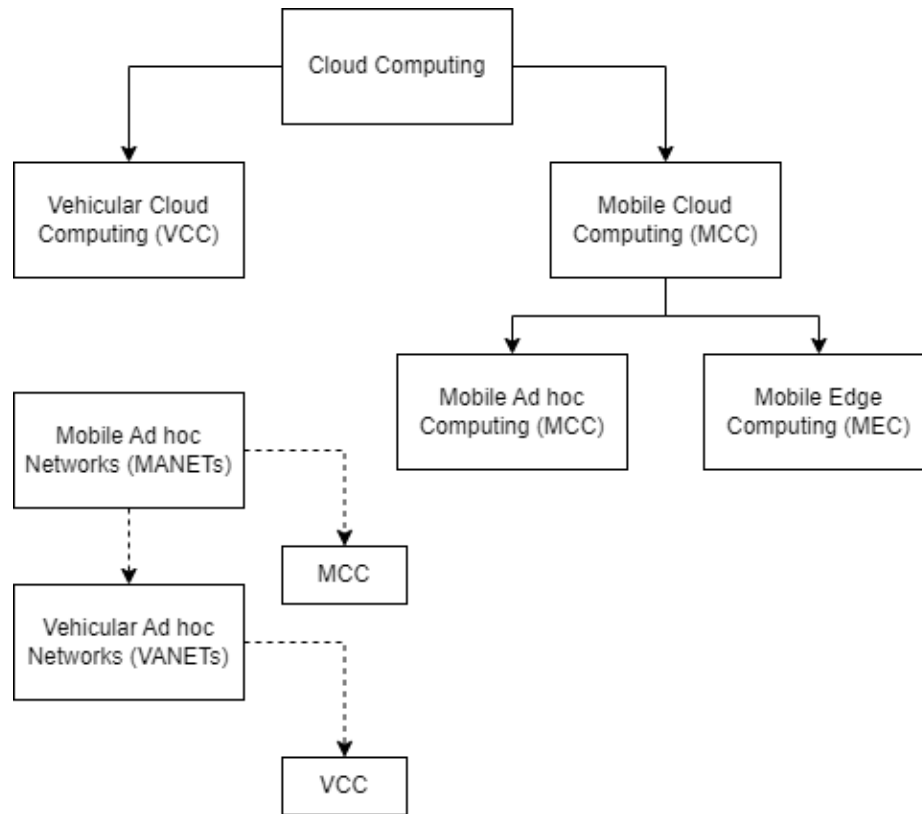


Figure 2.5: Evolution of vehicular cloud computing (VCC).

most instantaneously. Figure 2.5 illustrates the evolution towards vehicular cloud computing (VCC).

2.4 Vehicular Edge Computing

Vehicular edge computing (VEC) mainly integrates vehicles with edge devices. Its main goal is to place computational and cache structures in close proximity to the vehicles. In VEC, computational and storage resources, such as RSUs, mainly perform as edge servers close to vehicles for processing and storing data. Placing the edge servers close to the vehicles reduces communication latency and enables faster access to information. Figure 2.6 shows an example of a VEC architecture.

2.5 Vehicular Fog Computing

Day by day, the number of mobile devices and applications is increasing, and devices and vehicular applications are getting smarter. However, these resources can only

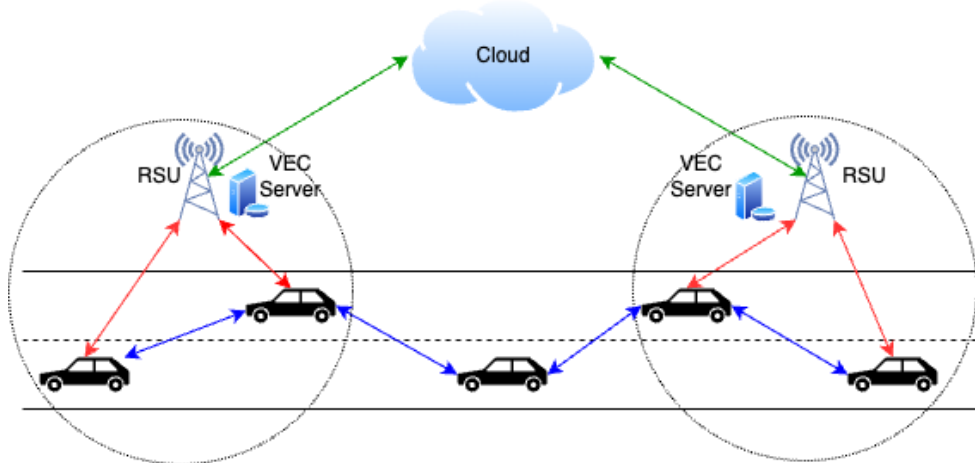


Figure 2.6: General vehicular edge computing (VEC) Architecture.

perform efficiently when it is being served with relevant data within a short period of time. Moreover, these applications require significant computing and communication capacity. By keeping that in focus, vehicular fog computing has been introduced. Vehicular fog computing provides the capability to move computing resources closer to the edge of the network. This functionality reduces the latency constraints and traffic overload of the vehicular clouds. This also refers to a decentralized system that brings computing resources and services to the edge where the data is being generated and acted upon [33].

Vehicular fog computing is a promising solution for addressing the increasing demand for real-time data and communication services in vehicular networks. For example, in a connected vehicle scenario, a large amount of data is generated and transmitted between vehicles, RSUs, and the cloud. This data includes navigation maps, safety alerts, vehicle performance data, and entertainment content. By bringing computing and storage resources closer to the vehicles and RSUs, vehicular fog computing can significantly reduce the latency and improve the reliability of data transmission.

One of the key benefits of vehicular fog computing is its ability to support real-time data processing and analysis. For instance, in a connected vehicle scenario, vehicular fog computing can process and analyze data generated by various sensors, such as cameras and radars, in real-time to provide valuable insights to the drivers and other road users.

Another advantage of vehicular fog computing is its ability to support scalable and cost-effective data storage and retrieval. By bringing storage resources closer to the vehicles and RSUs, vehicular fog computing can significantly reduce the cost of data

transmission and storage compared to traditional cloud-based solutions. Moreover, it can support scalable data storage and retrieval by providing a distributed architecture that can be easily expanded to meet the growing demand for data storage and retrieval.

2.6 Information-Centric Networking

Information-centric networking (ICN) is an approach to evolve the internet to directly support this use by introducing uniquely named data as a core Internet principle. As a result, data becomes independent of location, application, storage, and means of transportation, enabling in-network caching and replication. However, due to the dynamic nature of IoV in terms of a frequent network partition, differentiated mobility, and uneven nodal distribution, the present IoV suffers from many performance degradations, especially when the vehicles employ TCP/IP stacks as their addressing and traffic control criterion. In fact, during the brief connection period, the fast-moving vehicle may not have the chance to be given an address and establish end-to-end connections by several handshakes with other vehicles. As a result, information-centric networking (ICN) has quickly ascended to the spotlight and gained enormous popularity in IoV by its ability to address the content dissemination issue among different vehicles, i.e., requesters and consumers in ICN. As the name indicates, the ICN cares about the content” WHAT they need,” not” WHO provide” [31].

2.7 Vehicular Caching

Vehicular caching stores frequently accessed data closer to vehicles in a vehicular network instead of retrieving it from a central server or the cloud. The idea behind vehicular caching is to reduce the latency and network traffic associated with fetching data over long distances and improve the overall performance and reliability of the network.

Vehicular caching can be implemented in various ways, including in-vehicle caching, roadside caching, and cloud caching. In-vehicle caching involves storing data directly on the vehicle, while roadside caching involves storing data on servers near the road, such as in RSUs. Cloud caching involves storing data in a centralized location, such as a cloud data center, but bringing it closer to the vehicles and RSUs through edge computing.

Vehicular caching can be used to improve the performance and reliability of various

applications in a vehicular network, such as navigation, safety alerts, entertainment, and vehicle performance data. For example, by caching navigation maps and real-time traffic information closer to the vehicles, a vehicular network can provide drivers with faster and more accurate navigation services. In conclusion, vehicular caching is a crucial technology for improving the performance and reliability of vehicular networks. By storing the most frequently requested content nearer to the vehicles and RSUs, vehicular caching can reduce latency, improve data processing and analysis, and provide a more reliable and efficient experience for drivers and other road users.

Nowadays, caching is an essential strategy for an efficient content delivery technique to serve the huge number of requests from vehicles at the lowest possible time possible. Regarding vehicular networks and distributed systems, it certainly has to be implemented effectively due to the uncertain pattern of requests and high mobility environment. A well-implemented caching strategy can mitigate the network traffic load and reduce latency [24]. When a vehicle requests content, it can be served by a caching node rather than initially traced to the central server. Although it can serve the nodes adequately, the volume of the storage to cache is limited. Therefore, there are better plans to store large volumes and amounts of content in the cache. A caching strategy also implies which content to cache and how to place the cache and satisfy as many requests as possible. Therefore, designing an effective and efficient caching strategy that acknowledges the requests in high mobility vehicular network scenes to utilize maximum cache is worth a challenging research problem and, indeed, a critical task.

Chapter 3

Related Works

Caching is a technique used to temporarily store data in a fast-access location for quick retrieval. This reduces the source server's load and improves the system's overall performance by providing faster access to frequently used data. Though caching makes data more readily available and distributes it fast; still, cache management strategies have many challenges due to the dynamic nature of the large-scale vehicular network. Furthermore, data demand is increasing from time to time, and the nature of data communication has been evolving. In that regard, cache management strategies have also been extensively explored [7]. As a summary of recent approaches related to caching, we describe some notable recent works on cache strategies and their challenges.

3.1 Delay aware caching

Caching on the network's edge is the popular caching strategy to reduce information transmission delay. A delay-aware caching (DCC) algorithm proposed in [17] optimizes vehicle association, caching decisions, and pre-caching of RSUs to minimize the content fetching delay. This study focuses on reducing the content fetching delay in the internet of vehicles (IoV) networks. IoV networks allow vehicles to connect with other devices, infrastructure, and users. Services, content, and applications can be provided to vehicles and drivers through the mobile wireless network, but this causes high latency and low quality of service (QoS). Mobile-edge computing (MEC) is used to reduce latency and improve QoS by providing computing resources and storage capacity. Popular content can be stored in the MEC server, which is connected to the roadside unit (RSU), and sent to the vehicle via the vehicle-to-infrastructure (V2I) link. The available storage capacity of RSUs is limited, so the content caching

strategy must be optimized to reduce cache redundancy. The authors propose a delay-aware caching algorithm that optimizes the cooperation between vehicles and RSUs to minimize network content fetching delay. The algorithm takes into consideration the handover of vehicles during the content fetching process and reduces delay by pre-caching required content for handover vehicles.

3.2 Edge Caching

There are many edge caching strategies that have been researched recently. Zhao *et al.* presented a study that introduced joint server caches in a distributed way, where resource allocation is remotely distributed online to reduce response delays [38]. Even though there are relatively reduced response delays, the operation cost of vehicles imposes limitations in access and decision-making, requiring trade-offs. With the edge caching strategy and advancements in mobile computation power, approaches have utilized mobile devices as a significant part of caching solutions. Most recently, Yang *et al.* focused on streaming video and caching it in a partitioned group by establishing V2V communication between traveling vehicles. By proposing a group partitioned caching (GPC) algorithm, the approach considered cache hit ratio and latency to be the significant factors for the matrix of optimized caching strategy [37]. However, an edge server may lead to storage and bandwidth wastage due to continuing processing of the same content. To address this issue, E-Cache has been designed as a content caching strategy to address this issue [34]. It can predict service requirements based on a deep spatiotemporal residual network (ST-ResNet) and also optimize the execution time and consumption of the services related to the network.

3.3 Cooperative Service Caching

The edge caching prospect has given an advantage in reducing service time but lacks edge cooperation. Therefore, it cannot fully utilize the storage and computational capacities of the Edge resources. To address this issue, Ma *et al.* proposed cooperation between edge resources by addressing cooperative service caching among the edge and scheduling the workload [25]. Furthermore, Wang *et al.* introduced content request prediction by the popularity of the internet of vehicles to lower the cache hit ratio among the edge cache servers. This content request prediction has been formulated on a cooperative caching strategy and processes content among the cache nodes by content popularity requested by vehicles and RSUs [32].

3.4 Fog Caching

Due to the expansion of edge devices' computation power, modern network features have been shifting toward fog networks. Fog adds more functional capabilities and computational possibilities to this vast vehicular network. It has significant probabilities of caching strategies for reducing response time. In that regard, Hua *et al.* has featured a fog-based caching scheme on ICN that describes content caching closer to the edge nodes based on the content popularity. Also, the approach attempts to maximize the utilization of the caching nodes closer to the content delivery and integrated reactive caching, avoiding peak time congestion [16]. The work of Li *et al.* shows that a capacity-aware edge caching (CAEC) framework was introduced to efficiently cache contents by considering both the limited cache capacity of fog nodes and the primary center's connectivity capabilities [20].

3.5 Caching in 5g and emergence of AI

As information-centric networks (ICN) and edge computing dominate modern cloud-based mobile environments, the introduction of 5G has given a new dimension to the field. 5G-supporting networks and devices are available with more computational capability and greater storage capacity. Ullah *et al.* proposed an ICN-based edge computing capable of in-network caching strategy for 5G networks. It is a 5G radio access-based network enabling device-to-device communication with prefetching and caching based on content popularity that improves content distribution and performance of the device communication with the services [29]. A cooperative and collaborative caching strategy is also deployed in the 5G network using adaptive media cloud clustering and content distribution considering limited cache [27]. Modern technological advances in machine learning and artificial intelligence also have been integrated into caching mechanisms. Ning *et al.* has proposed using a deep reinforcement learning strategy dynamically construct edge computing and content caching enabling intelligent transportation systems to provide quality services with greater efficiency in the 5G-envisioned network [26]. Furthermore, Xu *et al.*'s work describe caching in a 5G-enabled mobile edge computing network with noted dynamic content caching and offloading. The work establishes a generative adversarial network (GAN) to predict the user demands effectively and based on the result of the GAN decision to be taken on both user demand and processing delay uncertainties [35]. More AI and machine learning opportunities empowered caching strategies in vehicular edge

Table 3.1: Summary of the related works.

Work	V2X	Conn	HM	CD	FC	CP	Model	Goal	Process
[17]	•	•				•	IOV	Reduce latency	DCC optimization
[37]		•	•	•		•	MEC	Max revenue	group-partitioned caching
[25]		•		•			MEC	Reduce response delay	Outer approximation
[38]		•		•			VEC	Reduce response delay	Joint service caching
[32]	•	•		•		•	IoV	↑ Cache hit ↓ Latency	LSTM request prediction
[20]		•			•	•	FCN	Maximize cache hits	Multi-class processor queuing
[16]		•		•	•	•	ICN	↑ Cache hit ↓ Latency	cluster based caching
★	•	•	•	•	•	•	FCN	Cache mng & distribution	Multi-layered fog caching

V2X: Vehicle to everything; Conn: Connectivity; HM: High mobility; FC: Fog computing; CP: Content popularity, CD: Cache distribution; IoV: Internet of Vehicles; MEC: Mobile edge computing; VEC: Vehicular edge computing; FCN: Fog cloud network;

computing [19]. Furthermore, edge caching with 5G enabled modern augmented reality applications, where virtual reality environments have also been mentioned as promising and practical applications [28, 21].

3.6 Remarks

High mobility, the dynamicity of IoV models, and low latency requirements for 5G networks give challenges for delay-aware caching. Edge caching reduces response delays, but it has storage, bandwidth waste, and redundant processing of contents. Therefore, cooperative service caching between edges has been explored. However, the service demand is known, but the service demand’s uncertainty has yet to be considered. Fog caching has enabled access to data in mobile environments quite substantially. Nevertheless, uneven user distribution in content delivery is a challenge that has yet to be solved. Our work is being proposed to meet high vehicle mobility to our advantage by contributing to the caching system and distributing the data across different zones in V2X and envisioned C-V2X fog networks. Table 3.1 summarizes the works related to our methodology.

Chapter 4

Problem Statement

Modern vehicles embedded with intelligent devices and applications have greater storage capacity and more computational capability. These applications and devices require continuous data feed to function properly. Moreover, nowadays, streaming data for mobile users is highly desirable. It is a crucial task to manage high traffic of data requests, has faster access to helpful information, and maintain the quality of service whether a vehicle is mobile or not.

However, it is challenging to meet all the demands while the vehicle continuously moves from zone to zone. The introduction of 5G can provide the capability to transmit data faster and maximize the quality of the service. 5G, the fifth generation of mobile networks, offers faster data rates and increased capacity than previous generations of mobile networks. However, its shorter coverage area is one of the challenges faced by 5G. The shorter coverage area means that the 5G network has a limited range, making it necessary to have more 5G cell towers in the same area to provide seamless coverage. As a result, frequent handover might be seen in the communication. Also, the implementation of 5G requires a significant investment in network infrastructure and maintenance, making it necessary to consider the trade-off between the advantages and disadvantages of 5G when deploying the technology in a specific area, forcing uncertainty over the area would be covered by the 5G network communication [1]. Figure 4.1 depicts a general scenario where all the components necessary for our layered design simultaneously co-exist in an urban center.

Consequently, vehicular mobility in the scene gives another situation and challenge to distribute data uninterruptedly in a limited time to a broader range. Besides, not all vehicles and devices are equipped with 5G technology urges us to have an alternative strategy. Hence, caching data is a compelling option to mitigate the abovementioned problems and provide uninterrupted mobile vehicle services. Per the

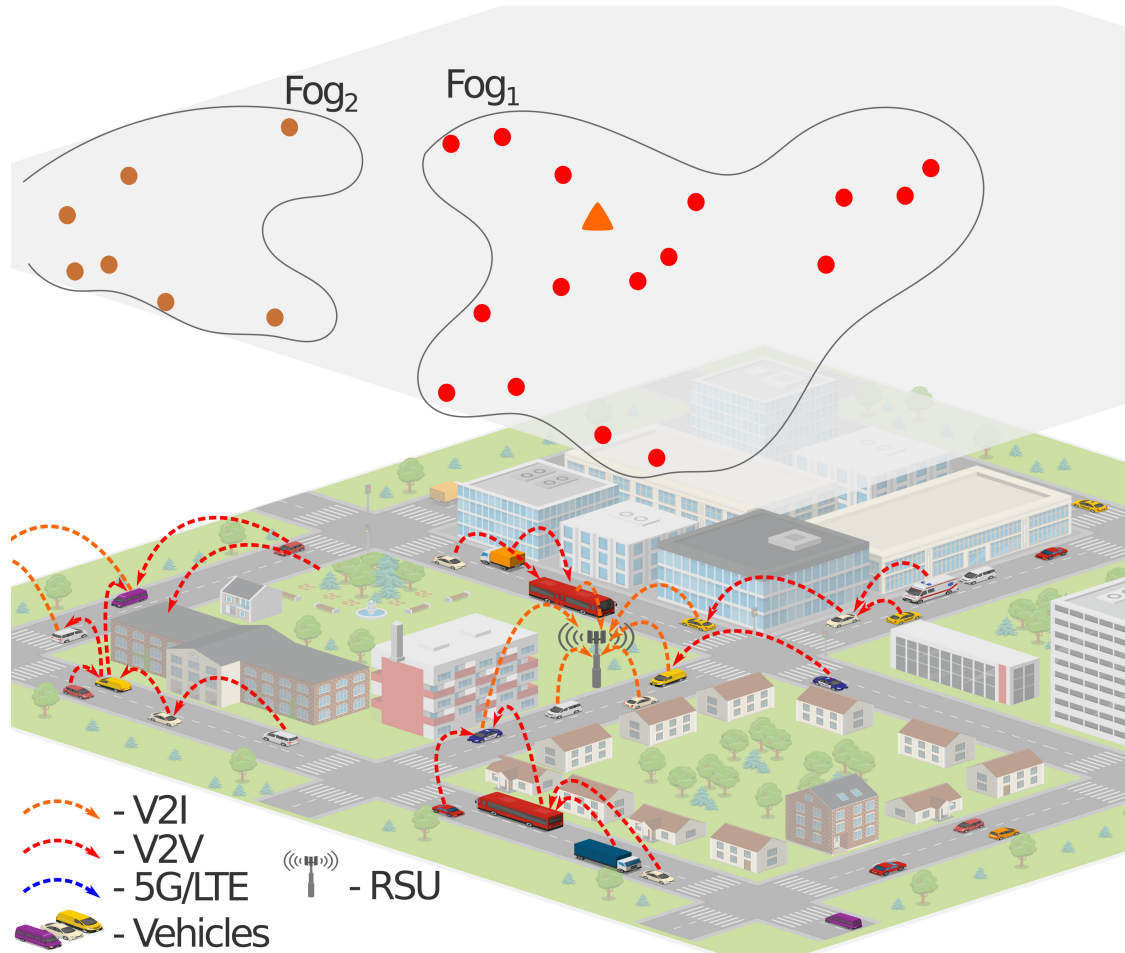


Figure 4.1: Scenario of fog regions in an urban center.

conventional caching design, if a cache miss happens, fetching data from data centers or clouds might derail the quality of service of the application requesting the service. At the time of cache miss, the delay in fetching the data from the data centers or clouds may abruptly the continuous service too. Although placing the cache as close to the edge devices may smoothen the data transmission, it will still require enough caching infrastructures within a shorter range.

In modern caching solutions, it is only sometimes required to fetch data from the data center or clouds due to the increasing availability of fog schemes and edge servers. Therefore, it is possible to utilize these storage units, which boost access to the cache faster. In order to maintain the high traffic of requests and access requested content faster, a layered caching strategy can be helpful. Therefore to reduce the cache miss and provide continuous data dissemination service in vehicular networks, an improved caching strategy will become necessary to support numerous intelligent applications and devices for modern vehicles. Moreover, to deal with the high mobility scenario,

mobility can also be used at our supremacy by delivering the important data to the cache of the mobile nodes as a transient cache. A transient cache is an additional adaptive caching location where the caching contents are contributed by the vehicle mobility and also used to deliver the popular contents among the regional interests.

Chapter 5

Hierarchical Transient Caching

Our proposed design focuses on a multi-level caching capability and distribution with vehicular mobility in vehicular fog environments. The caching system is comprised of caching spread across different points of the environment: vehicles and Fog controller multi-layers. The system also defines cache flow management strategies, data distribution approach, and cache coordination policies.

5.1 Multi-Level Caching

Our primary focus is to make data more readily available within a shorter time. Thus, having more caching nodes available to the requesting nodes is vital. Therefore, our design mainly has three-level caching nodes: primary cache, fog cache, and edge cache.

5.1.1 Primary Layer

The primary layer majorly consists of the cloud or data centers where actual data and information can be found. Even though data centers can handle a large number of requests and store lots of data, increased network traffic demands more data updates and requests which leads to longer response times and increased latency, which can potentially jeopardize the quality of service, particularly in areas that are far from the data centers. The remoteness of certain regions from the central data centers means a need for more localized and distributed computing infrastructure to support the increasing demand for high-speed and low-latency data access. Consequently, the response to the request may delay fetching from this layer. This layer is the last resort in fulfilling queries for data requests to feed into the network for implementing

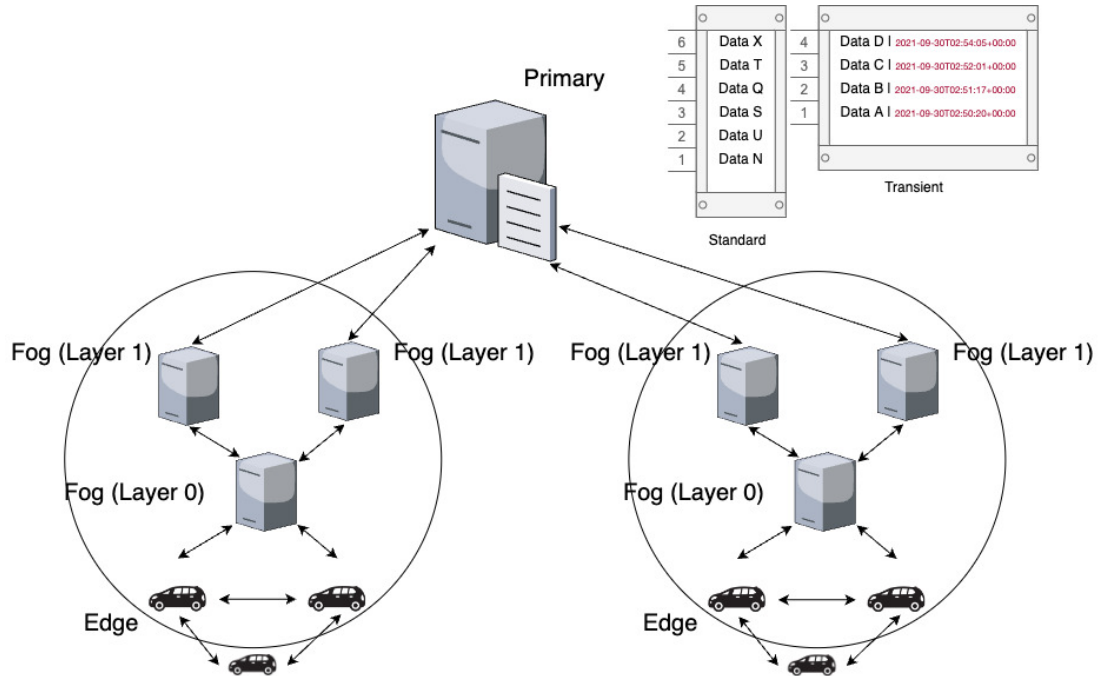


Figure 5.1: Abstract view of the architecture of our approach.

the layered cache structure. As a result, the fewer data is queried, the faster the data response is. If a request reaches the primary layer, the caches of the bottom layers must be updated immediately.

5.1.2 Fog Layer

The fog layer consists of fog nodes, such as gateways, access points, routers, and fog servers. It is situated in-between the edge layer and data centers. It also can be a hop distance from the edge devices. Fog nodes provide services to edge devices. Fog nodes are capable of complex computing, faster data transmission, and storing data. Fog nodes and cloud data center connections are enabled by the IP core networks, providing interaction and cooperation with the cloud for enhancing processing and storage capabilities. Fog nodes restrict the large volume of data sent to the clouds, reducing significant bandwidth and improving response time. Our proposed model assumes that this layer of fog nodes can handle many data requests and provide data processing prowess. Even though this layer can manage significant cache requests, it has a binding related to a physical location. Therefore, the caches may not always be available at the boundary of the moving vehicle.

Figure 5.1 shows an abstract view of our approach. This layer inhibits the complex

computing capability of the terminal devices. Data generated from the edge layer is either processed on the device directly or transmitted to a server or cloud. This layer can also temporarily store data, which is very useful to edge devices for caching. The data in this layer are closer to the edge devices and can feed data almost instantaneously. Therefore, it is necessary to place only the most relevant data in the cache to reduce the cache miss ratio and generate a faster response to the end device. One of our goals is to use vehicular mobility to our advantage. Vehicles moving from one zone to another can have completely new or significant data in their local storage. We place this data on our temporary cache, named *transient cache*, and include a timestamp on each cached data to track the importance of the data in terms of the time domain. This cache should not be mixed with the original cache placed in this layer which may lead to more cache misses instead of lessening them. Thus, it is vital to identify relevant data brought through the mobility of the vehicles so that they be cached separately and temporarily.

5.1.3 Edge Layer

This edge layer consists of endpoint devices and terminal nodes. The various sensors, devices, and applications embedded in the vehicle contribute to this layer. These devices request data from the fog layer and can provide valuable data to cache into the edge layer. It can communicate and exchange information through V2V communication. Therefore, caches situated in vehicles can collaboratively serve other vehicles within close distances, not necessarily requiring data requests to the fog nodes every time data needs to be fetched. Moving vehicles can simultaneously stream data and provide other data to the cache, which another device or application can request.

5.2 Cache Flow, Update, and Management

As discussed, one of our main objectives is to explore the possibility of vehicular mobility to our advantage, which is storing relevant cache contents brought by vehicles. Therefore, we stored cache data with timestamps. Timestamped cache data is necessary to update caches regularly. Keeping the primary layer's cache unchanged, we investigate the fog layer's dynamic cache. The fog layer cache needs are maintained in two separate adaptive parts. One is the original cache, and the second is the transient cache. The transient cache is added through the mobility of the vehicles and their embedded applications. The fog layer is also capable of forming multiple

Algorithm 1: Fog Layer Data Flow

Data: $F_i \leftarrow$ Fog nodes in the range; $F_h \leftarrow$ Number of Fog layer hops;
 $F_{limit} \leftarrow$ Limit of Fog layer Hops; $C_H \leftarrow$ Cache Hit Flag; $R_u \leftarrow$
Requested User; $fogDestination \leftarrow$ Cache store from upper layer

Result: *CacheHit*

```

1 if  $F_h \neq F_{limit}$  then
2   if  $F_i$  is  $fogDestination$  then
3     Store cache received from upper  $F_{i'}$  layer
4   else
5     if  $\neg C_H$  then
6       if  $F_i$  has the content then
7          $C_H \leftarrow$  True;
8         Forward message to  $R_u$  from  $F_i$ 
9       else
10         $F_h \leftarrow F_h + 1$ ;
11        Forward message to  $F_{i'}$  from  $F_i$ 
12      if  $F_h == 0$  then
13        Set  $fogDestination = F_i$ ;

```

layers. Whenever a request arrives from a vehicle/edge node, the fog layer creates a fog region with other fog-caching nodes within its perimeter. The closest fog nodes are noted as the bottom layer (0), and others within the range are the top layer (1). If the bottom layer cannot satisfy the request and the top layer does, then the cache is placed on the bottom layer with the response to serve future requests from closer proximity and reduce delay. Algorithm 1 shows the message flow between the fog bottom and top layers. The vehicles moving from one zone to another have relevant data that is new to the incoming zone and can be cached. This data is requested by another vehicle, and it can reduce the additional request to fetch from the primary layer.

Moreover, the transient section is continuously updated with the data from the incoming vehicles. For instance, a vehicle enters the zone and has data A, which is new to the fog server of that zone. Therefore, data A is stored in the transient cache with a timestamp. After some time, a new part of data A arrives in the zone with another vehicle. We compare the timestamp of both the data and can tell the vehicle that the server has a newer part of the particular data A and provide it to the vehicle. In that way, a vehicle does not need to query the primary layer to fetch the data, which may cause a more significant latency issue. We update the cache with the

Algorithm 2: Edge Layer Data Flow

Data: $R_i \leftarrow$ Resource in the range; $R \leftarrow$ Range of the resource; $N_h \leftarrow$ Number of Hops; $N_{limit} \leftarrow$ Limit of Hop count; $C_H \leftarrow$ Cache Hit Flag; $R_u \leftarrow$ Requested User;

Result: *CacheHit*

```

1 for all  $R_i \in R$  do
2   if  $\neg C_H$  then
3     while  $N_h \neq N_{limit}$  do
4       if  $R_i$  has the content then
5          $C_H \leftarrow$  True;
6         Forward message to  $R_u$  from  $R_i$ 
7       else
8          $N_h \leftarrow N_h + 1$ ;
9         Forward message to  $R_{i'}$  from  $R_i$ 

```

newer version if the transient cache is older than the incoming data. For simplicity, time aware least recently (TLRU) is a cache eviction strategy that discards the least recently used items when the cache reaches its capacity. Unlike the traditional Least Recently Used (LRU) algorithm [3], TLRU has a time limit set for each cache item, beyond which the item is removed from the cache. This algorithm is often used in systems where the data becomes stale after a certain time period [4]. TLRU has been placed as the cache eviction policy for the transient cache. Every content has a certain lifespan and is updated based on the timestamp of the least recently used content. We also follow up on a significant cache factor and observe whether data are requested more often. If devices are interested in a particular data entry in the transient cache through cache requests, the system moves that data entry from the transient to the original cache. Algorithm 2 refers to the communication between vehicle to vehicle.

5.3 Distribution

Our proposed caching model promotes data distribution across a large-scale urban environment using vehicular mobility through a transient cache. In previous works, we found that caching popular content closer to edge devices is a method to improve content delivery efficiency [17]. It is also implemented to arrange the most recent, popular, or trending data in a particular area and place that data in the cache. Thus, we have explored transient cache distribution while vehicles request from the fog

Algorithm 3: Cache distribution

Data: $D_r \leftarrow$ Data request from V_0 ;
 $D_f \leftarrow$ Fetched Data;
 $D^* \leftarrow$ Data to distribute;
 $msg \leftarrow$ Transmitted message;
 $TransientMap \leftarrow$ Transient Cache;
 $StandardMap \leftarrow$ Standard Cache;
Result: $TransientMap$

```

1 Function distribute( $msg$ ):
2    $D_r \leftarrow msg.getRequest()$ ;
3    $D^* \leftarrow TransientMap.top(n)$ ;
4   if  $D_r \in TransientMap$  then
5      $D_f \leftarrow TransientMap.get(D_r)$ ;
6      $TransientMap.update(D_r, now())$ ;
7   else
8     if  $D_r \in StandardMap$  then
9        $D_f \leftarrow StandardMap.get(D_r)$ ;
10       $TransientMap.insert(D_r, now())$ ;
11     else
12        $sendRequest(msg)$ ;
13    $msg.setResp(D_f)$ ;  $msg.setRcvr(V_0)$ ;
14    $msg.setAdditionalData(D^*)$ ;  $sendResp(msg)$ ;

```

layers. While a vehicle is leaving the range of a fog region or zone, it carries in its local cache data entries that can be distributed to the upper layer caches of other zones. Moreover, it can be a new data item to that particular zone, placed in its transient cache, and distributed to another vehicle. Algorithm 3 shows the steps for placing the idea of distributive cache to the transient cache.

5.4 Policies

The cache storage capacity is constrained, and end-user devices must grant authorization to services for the application's memory space to be leveraged effectively. Consequently, policies must be bound with a strategy for updating and maintaining the caches in the system at different layers. The fog layer cache is adaptive, and we cannot change the original cache of the fog layer. Constant changes and updates in this cache can lead to misleading data entry updates. Due to the movement of vehicles, continuous data flow from the edge layer arrives in the fog layer, and it can cause more cache misses than hits. Therefore, a separated transient cache area

Table 5.1: Undertaken policy parameters.

Parameter	Value Range
Maximum Hops	2 – 3
Caching Content (Max)	200KB
Caching Capacity of Fog Nodes (Appr.)	2MB
Caching Capacity of Edge Nodes (Appr.)	1MB

is implemented. It must be adaptive to adapt to the request traffic with the cache dynamically. The user storage we want to use for distribution must be permitted and should be as minimum as possible. Policies can be implemented to exchange the caches between original and transient in the fog layer. These are some policies that might be needed to implement the strategy. Table 5.1 shows the undertaken policy we have taken for our experiments.

Chapter 6

Popularity-based Distribution

The multi-layer caching strategy places the cache closer to the point of use and makes data more readily available. An efficient caching strategy institutes the proper distribution of the most valuable data regarding trending, frequent, or popular contents in the cache. Although caching speeds up data access, it has a minimal capacity. Caching popular content can improve performance by reducing the number of requests to the server and the amount of data that needs to be transferred over the network. Popular content can include images, videos, text, and other types of data that users frequently access. Usually, in large mobile networks, popular content is stored on a network of servers located in different geographic locations, which allows the content to be quickly delivered to users based on their location.

6.1 Fog Zone Oriented Popularity Caching

Fog nodes, which are nodes located at the edge of the network, are used to cache popular content and serve it to the nearby requester without having to send a request to the main server. This reduces network latency and improves the user experience for devices that are connected to the fog node.

In our fog computing strategy, popular content caching is used to improve performance and reduce the load on the fog server by storing frequently accessed content closer to the vehicles.

6.1.1 Local and Global Popular Contents

Local popular content refers to popular content within a specific geographic area or region. In contrast, global popular content refers to popular content across a larger,

global audience. Local popular content is specific to a particular network or region and is based on the usage patterns and preferences of the users within that area. Here are some examples of local popular contents in the context of vehicular networks:

- **Maps and navigation data:** This includes up-to-date map data, real-time traffic information, and routing information, which are essential for safe and efficient navigation in an urban environment.
- **Local weather and road conditions:** Vehicles need real-time information about the weather and road conditions in their immediate vicinity to avoid hazards and drive safely.
- **Local parking information:** This includes real-time information about available parking spaces, parking rates, and parking restrictions, which can help vehicles find parking quickly and efficiently.
- **Points of interest (POIs):** This includes information about local attractions, restaurants, hotels, and other destinations that drivers may be interested in visiting.
- **Local news and events:** This includes information about local news and events, such as road closures, roadworks, and traffic disruptions, which can help drivers avoid traffic congestion and delay.

Global popular content refers to items that are popular and widely accessed across multiple networks and regions. These items tend to have a broad appeal and are often associated with popular trends, cultural events, or breaking news stories. Examples of global popular content include blockbuster movies, viral videos, and popular music.

Caching local popular content helps reduce network latency and improve the user experience within a particular region while caching global popular content helps to reduce the overall load on a system. Whether to use a local cache or a global cache depends on the specific use case and the system's requirements. Our system has several use cases which have led us to enumerate local cache within the fog nodes.

- The content request is dynamic and changes frequently, and it would be costly to update the content on a global cache.
- Network latency is a concern, and it is necessary to reduce the distance between the user and the source of the content.
- Utilize the common interests of the high mobile vehicles in a specific geographic area or zone.

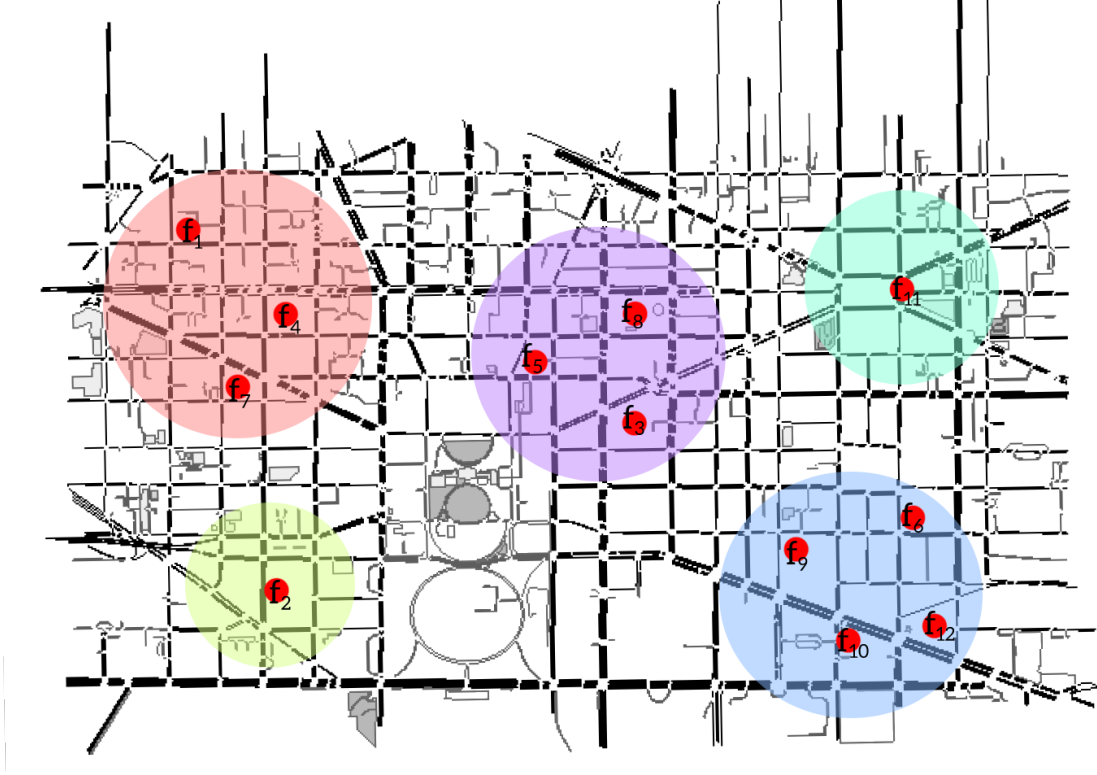


Figure 6.1: Zone formation with fog nodes (Red dots).

It is demanded that the cache should be categorized according to the most popular items. In our system model, we placed the cache according to the zone adhered to. For example, the transient cache of fog controllers is categorized to the zone to which the mobility of the vehicles contributed it. Each item is the most frequently requested data from that particular zone.

6.1.2 Zone Formation

The zone formation relies on the fog nodes' multi-layer structure. The fog controller initiates a self-request message to identify other fog nodes that can reach within its perimeter to create its sub-layers. It ultimately defines its zone by denoting the zone roles. Vehicles within the perimeter of a particular zone are served through wireless communication. Figure 6.1 depicts the zoning of the multi-layered fog nodes.

A randomly selected fog controller initiates a self-message to start the process of creating the roles of its particular zone. The self-request message, C_{msg} , is scheduled with a text, T_{msg} , that notes other fogs to participate in the zoning. Each cluster maintains a vector of fog information that falls into the same multi-layer structure, V_f . If the scheduled message from fog controller f_s is received by fog, f_n , where

Table 6.1: Fog Nodes Role Allocation.

Zone	Node	Layer
	f_1	0
1	f_4	1
	f_7	1
2	f_2	0
	f_5	0
3	f_3	1
	f_8	1
4	f_{11}	0
	f_{10}	0
5	f_9	1
	f_{12}	1
	f_6	1

$n = (1, 2, \dots)$, then certainly that particular fog, f_n , is in the perimeter of f_s . The fog is being denoted its role as $(zone_{id}, layer_{no}, fog_{id})$. Finally, with the information of f_n being added to the vector, V_f . A fog node of layer-1 can fall under multiple zones due to the dense geographic location of the fog. In such circumstances, the fog node of layer-1 distance closest to layer-0 will be added to that particular zone.

While the self-message is reached to a vehicle, the information of that particular vehicle is also recorded in a table respective to the zone id to identify the vehicles connected to the region. Vehicle speed, direction, position, and timestamp are recorded and kept in the table. Table 6.1 shows the roles assigned for particular zones in our system.

For example, the f_1 controller starts the initiation of zone 1 and reaches the f_4 node. As a result, f_4 , situated within the first hop of f_1 , identifies as layer 1, and it is assigned to zone 1. Furthermore, the message reaches out to f_7 , which is also within the first hop to f_1 , and thus it is being assigned to layer 1 of zone 1. Moreover, the zone map is being created all over the network. Algorithm 4 our zone formulation algorithm.

6.1.3 TLRU Factor Function

We have used TLRU as our replacement policy in our model for transient caching. The difference between LRU and TLRU is that time to use (TTU) provides a timestamp of the items in the cache based on their locality and publisher's requirement. In

Algorithm 4: Fog Zone Formulation

Data: $C_{msg} \leftarrow$ Scheduled Self Message;
 $T_{msg} \leftarrow$ Transmitted message to zoning;
 $H_{limit} \leftarrow$ Number of Hop Limits;
 $F_{info} \leftarrow$ Self message recipient fog info object
Result: Vector of fog nodes, V_f

1 **Function** handleLowerMsg(C_{msg}):
2 **if** $C_{msg} == T_{msg}$ **then**
3 **if** $C_{msg}.getFogHopCount() \leq H_{limit} \ \& \ F_{node} \notin V_f$ **then**
4 $F_{info}.setZone(C_{msg}.getZone());$
5 $F_{info}.setLayer(C_{msg}.getLayer() + 1);$
6 $F_{info}.setPosition(curPosition);$
7 $F_{info}.setFogId(getCurrentNodeId());$
8 Add F_{info} to V_f
9 $C_{msg}.setFogHopCount(C_{msg}.getFogHopCount() + 1);$
10 $sendDown(msg);$

other words, it does provide the ability to generate more local-centric usefulness for the items to be cached. Also, it provides more control to local zone admins to administrate the local network storage, which is a very suitable eviction policy for fog-based and ICN-based network environments.

TLRU calculates TTU for the item cached based on a composite function defined as the TLRU factor function. Local policies and user requirements prior to being set within the factor function to administer the TTU. The composite function ($f \uparrow g$) is defined such that function f calculates the worth of the content by the content size. The larger content minimizes worth, while smaller content maximizes the worthiness of the item to be cached. Meanwhile, function g formulates the worthiness of the request frequency of the item and if the data is requested from an adjacent zone. Finally, Algorithm 5 generates the time to use of the particular content by measuring the total worthiness.

$$TTU = (f \uparrow g) \quad (6.1)$$

$$f(x) = \gamma * Time_{max} \quad (6.2)$$

$$g(y) = \beta * Time_{max} + Time_{buffer} \quad (6.3)$$

Algorithm 5: Calculate TTU

Data: $record \leftarrow$ Content to be cached;
 $size \leftarrow$ Size of the content;
 $worth \leftarrow$ Calculation of worthiness of content
Result: Time to use, ttu

```

1 Function calculateTTU( $record, zoneId$ ):
2   if  $record \notin T_{cache}$  then
3      $size \leftarrow sizeOf(record)$ ;
4      $worth \leftarrow calculateSizeScore(size)$ ;
5      $worth \leftarrow worth + calculateZoneScore(zoneId)$ ;
6   else
7      $worth \leftarrow getCurrentTTU(record)$ ;
8    $ttu \leftarrow worth + (rand() * Time_{buffer})$ ;
9   return  $ttu$ ;

```

Where $f(x)$ denotes the calculation of worthiness based on the size of the content, γ is a multiplication factor related to the size. In this study, three levels of multiplication factors have been accounted for as (0, 0.5, and 1). Therefore, greater-sized content will be worth less time value, whereas mid and smaller-sized content will be worth much because of the lightness of the content.

Also, $g(y)$ accounts for the calculation of the worthiness of the zone that the content originates from. The multiplication factor $\beta = (0.25, 0.5, \text{ and } 1)$ is associated with the frequency of the contents are requested from the zone originated. Content from the hot zone is worth to be cached due to the high probability of the frequency at which the content might be requested. $Time_{max}$ is dynamically adjusted, keeping the observation of the traffic load of the system during a specific interval of time.

6.1.4 Popularity Threshold Periods

Our system uses a popularity index table to track and measure content's popularity based on how frequently they are accessed or used. The popularity of a piece of content is determined by counting its frequency of access and comparing it to a set threshold value, ΔT . If the frequency of content exceeds this threshold, it is considered popular and is added to the popularity index table.

However, to prevent the same popular content from being selected over and over again, a freeze time denoted as δT , is introduced. During this time, the frequency of popular content is reset to a default value, allowing for a more accurate representation of its current popularity.

Table 6.2: Transient cache structure.

Index	Content	TTU	Zone	Frequency
1	10111101	200	2	1
2	11010001	120	2	0
3	11101100	110	2	0
4	11100000	100	5	2
5	01010101	500	5	0
6	00011001	50	3	0
7	10111001	180	3	1
8	01001100	110	3	1

The freeze time of popular content also significantly impacts its time to use (TTU), which refers to the amount of time it remains in the cache before being removed. A longer freeze time will increase the TTU, making the content more likely to remain in the cache for longer. On the other hand, a shorter freeze time will decrease the TTU, making the content more likely to be evicted from the cache sooner. Table 6.2 visualizes the structure of our map object of transient cache inherited with zone and frequency in fog nodes.

6.2 Inter Fog Popular Content Distribution

Regarding the popular content distribution, we have utilized our mobile nodes to distribute the caches from zone to zone while placing the important data in their response to the requests. However, in terms of fog node cache, we formed a cooperative effort to have the most frequent or trending content across the fog layer tiered cache nodes at a specific interval time. While request traffic is relatively low during the freezing period, we distribute the caches between the layered fogs. Hence, the cache from the bottom layer is distributed to the upper layer, and thus, it is updated with popular content. The delivery delay could be reduced during high traffic intensity. Algorithm 6 distributes the top caches between the layers of a zone.

The popular cache selection starts from the bottom layer. The cache is being transferred to the upper layer. The upper layer fog nodes select the top contents in their transient cache. The upper layer fog nodes then find the unique cached contents from its storage and transmit them back to the bottom layer. The bottom layer cache is then refreshed with the incoming cached contents from the top layer.

Algorithm 6: Inter fog cache distribution

Data: $C_{msg} \leftarrow$ Scheduled Self Message;
 $T_{msg} \leftarrow$ Message to cache distribute;
 $T_{cache} \leftarrow$ Cache Content;
 $M_f \leftarrow$ Map of fog nodes;
 $f_{id} \leftarrow$ Current Fog Id;
 $F_{info} \leftarrow$ Self message recipient fog info object
Result: None

- 1 **Function** handleSelfMsg(C_{msg}):
- 2 **if** $C_{msg}.getText() == T_{msg}$ **then**
- 3 **if** $f_{id} \in M_f$ **then**
- 4 $T_{cache} \leftarrow$ Fetch top n caches;
- 5 send(T_{cache});

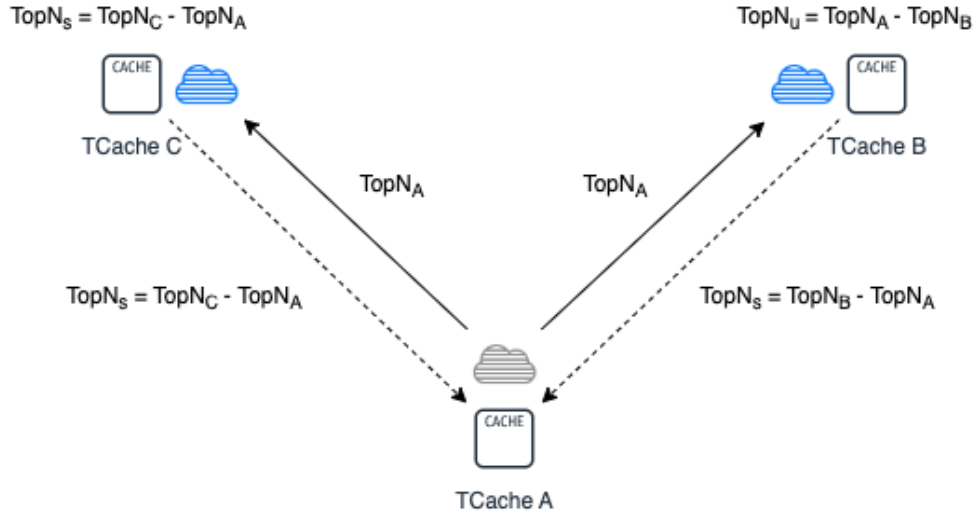


Figure 6.2: Inter fog top cached content distribution.

As example, at freeze period, δT , n^{th} layer top cache is being send to $(n+1)^{th}$ layer. $(n+1)^{th}$ layer identifies the unique top contents and keeps them in its transient cache. After completing the process, $(n+1)^{th}$ layer fog node sends back its top content to the n^{th} layer fog node. Similarly, n^{th} fog node identifies the unique contents and keeps them in its transient cache. Figure 6.2 shows the flow of popular content distribution between the fog layers.

6.3 Cache Load and Distribution

The vehicles leaving out the zone distribute the transient cache of that particular zone. Our system implies vehicle recording, which has the direction and position of the vehicle. The recording is updated at a specific watch time of θ . From the information, we predict whether the vehicle is heading toward a particular zone or not. The prediction of a vehicle to be crossing a zone is calculated through linear regression analysis. The popular content is being placed with the response to be distributed using vehicular mobility. While the top cache has been fetched, any content from the zone the vehicle heading is available, and whether the content is more frequently requested is observed. If available, the content is placed in the transient cache of the vehicle.

Chapter 7

Performance Analysis

Simulation has been used to observe the behaviour of our proposed model and its effectiveness in a complex scenario. Additionally, simulation in a highly strained configuration validates the effectiveness of our methodology and objectifies to gather performance metrics from each execution. The simulation is being conducted thoroughly in an urban realistic computing simulated environment because it mimics real-world conditions, such as traffic congestion, road networks, and communication patterns. The simulation results in an urban scenario can provide insights into how proposed our mechanism will behave in real-world deployment and help identify any potential limitations, issues, and improvements. The vehicles in the simulation are capable of maintaining communication through both V2V and V2X communications.

7.1 Simulation Setup

The simulations were built using Veins 5.1 [12], an established open-source framework for vehicular networks. The transmissions are supported through OMNet++ 5.7 [30] with INET-Framework v4.2.6 to simulate V2X communications between vehicles and fogs along with Veins. SUMO 1.8 [23] is an open-source, microscopic, highly portable and continuous traffic simulation package designed to handle large networks and vehicular mobility.

Veins is an open-source simulation framework for vehicular networks. It enables researchers and developers to study and evaluate the performance and behavior of various applications and protocols in a realistic and controlled environment. Veins supports the simulation of different types of vehicular networks, including Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Everything (V2X) networks. Veins is based on the widely used simulation tool OMNet++, and it inte-

grates a number of existing technologies and standards, such as SUMO (Simulation of Urban Mobility), INET (Internet Network), and 802.11p (Wireless Access in Vehicular Environments). This integration enables Veins to simulate the complex and dynamic behavior of vehicular networks, including mobility, communication, and applications.

OMNeT++ is an open-source discrete event simulation framework designed to model, simulate, and analyze complex network systems, such as communication networks, computer networks, and transportation networks. OMNeT++ provides a comprehensive set of simulation tools, including a simulation kernel, a graphical user interface, and a set of libraries and tools for modeling, simulation, and analysis. One of the key features of OMNeT++ is its modular architecture, which enables users to easily create and extend simulations by building and integrating simulation models. In addition, OMNeT++ supports a variety of modeling languages, including C++, and NED (OMNeT++ Network Description Language), making it flexible and easy to use for a wide range of simulation tasks. OMNeT++ has been widely used in academia and industry for research and development in various fields, such as telecommunications, computer networks, transportation, and energy. It is widely recognized for its reliability, scalability, and ease of use, and it has been used in numerous large-scale simulations and experiments.

SUMO provides a flexible and user-friendly interface for creating and editing road networks, defining traffic scenarios, and configuring simulation parameters. It also supports a variety of output formats, including animations, graphs, and statistics, allowing for detailed analysis of the simulation results. SUMO has been widely used in a range of applications, including transportation planning, traffic management, and traffic engineering. It has also been used to evaluate the impact of various traffic management strategies, such as traffic signal control, road pricing, and demand management, on traffic flow and congestion.

7.2 Evaluation of Multi-Level Caching

We evaluated our multi-level caching structure through simulation and performance comparison in this section. We aim to simulate the environment by setting up the parameters and metrics and determine how our approach performs against other protocols. Also, finding out how much impact transient cache generates integrating our the multi-level caching topology.

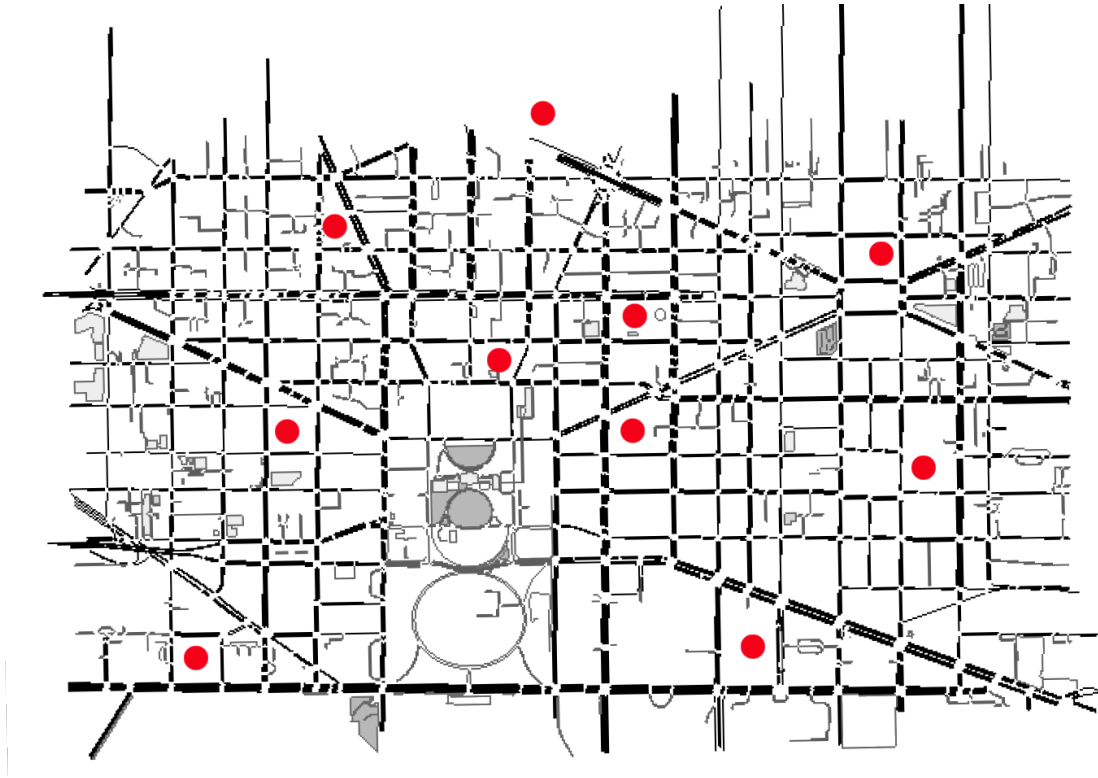


Figure 7.1: Map of simulated scenario: Washington DC, USA.

7.2.1 Simulation Scenario

The simulation scenario is designed based on an urban area of Washington DC, USA, as shown in Figure 7.1, where the area map size is $3088 \times 2517m^2$. The area consists of complex junctions, edges, and obstacles, generating a realistic simulation environment. The red dots indicate the position of RSUs.

7.2.2 Simulation Parameters

We have analyzed different vehicle densities ranging from 20 to 100 vehicles moving simultaneously. The vehicle's moving speed is from 0 to 50 km/h, considering traffic jams or urban area speed limits. The vehicles are generated independently in the simulator. Consequently, the source and destination route points are random, and each seed for a simulation run generates a different route. However, the vehicle trip from point A to B is fixed in the route file. The content requests originate from the moving vehicles. A vehicle request content every k sim time. The vehicle interface keeps a stack of content requests. Contents are picked to be requested in consecutive order. The content distribution to start the simulation has been set to initial (3-5)%

Table 7.1: Simulation parameter settings.

Parameter	Value Range
Simulation time	2500s
Washington DC	3088 x 2517m ²
Vehicle density	20 – 200
Vehicle Speed	0 – 50km/h
Fog Controller density	2 – 15
Cache Replacement Policy (Transient)	TLRU
Cache Replacement Policy (Standard)	LRU
RSU PHY model	IEEE 802.11p
RSU communication range	300m
Vehicular communication range	300m
Transmission power	20mW
Number of iterations of experiment	10 – 20
Confidence interval	95%

of the whole network. Some parts of the contents are distributed across the network for vehicles to find the requested data; otherwise, randomly generated data over the network does not cause cache hits. As an example, the whole network has a set of data objects [1-1000] randomly distributed to the set of fogs. Then a vehicle can request within a set of [1-1000] data objects with the content distribution set to (3-5)%. The higher distribution of the contents means the chances of requesting the same data objects to be frequently chosen to request.

In our experiments, we used different numbers ranging from 2 to 15 of fog controllers to measure the impact of the density of fog on the cache of multi-layer fog servers. Each roadside unit (RSU) has a communication range of 300 meters and the IEEE 802.11p protocol to connect to the vehicles. In our simulations, each RSU in the environment defines a Fog controller, which allows the Fog servers to communicate with the vehicles surrounding. Our simulation has at least ten and up to 20 iterations per point of metric generation. The confidence interval has been set to 95% to provide a more accurate average value and idea of the result analysis.

For cache replacement policy, we have used time aware least recently used (TLRU) [4] for Transient Caches. TLRU is a variant of LRU, but the stored content has a valid lifetime. As the contents in the transient cache are temporarily stored, it is necessary to have the contents with a good life span. Therefore, we placed a widely used, least recently used (LRU) content replacement policy for standard caches. Table 7.1 summarizes our simulation parameters.

7.2.3 Performance Metrics

In our experiments, we noted the following performance metrics:

- **Cache Hit Ratio.** Cache hit ratio is the measurement of how many requests is satisfied by the caching nodes, comparing the total number of requests the node receives. The total number of requests denotes the number of requests satisfied and failed by the node. If a caching node receives a total of R_T requests and it satisfies R_S and fails R_F requests, then Cache Hit Ratio (CHR) would be,

$$CHR = \frac{R_S}{R_T} = \frac{R_S}{R_S + R_F} \quad (7.1)$$

- **Latency.** Latency is the time it requires to serve the content to the requester from when it was requested. In order to calculate the latency, we need the request forwarding time and response time to approach the original request node. In that case, if the request node and the delivery node are v_a and v_b respectively, therefore T_f , the total time taken to forward the request to the delivery node and T_b , the total time taken to backward the request to the requesting node from the requesting node, then latency L should be,

$$L = T_f(v_a, v_b) + T_b(v_b, v_a) \quad (7.2)$$

We need to measure the average latency to testament the overall experiment, where R is the number of measured requests.

$$L_{avg} = \frac{\sum L}{R} \quad (7.3)$$

7.2.4 Protocols

Our performance analyses compared three protocols through the simulations:

- **Multi-TC.** Our proposed protocol uses a multi-layer architecture and implements transient caching across the Fogs in the urban center.
- **Multi-WTC.** This protocol represents the multi-layer design *without the transient caching* in the place where caches are distributed across Fog controllers and nodes (vehicles). The cache replacement policies in this protocol follow the LRU approach. We expect to observe it to enable faster access to data from the closer cache among vehicles and Fog controllers. However, it does not match the distributive nature of mobility demands from the vehicles.
- **CBS.** This fog cluster-based caching architecture represents a recent related

work [16] described in the previous sections. This work utilizes the requests from user groups and forms a fog cluster with multiple tiers of caching nodes within preset ranges. If the content exists in the t^{th} tier, then the content is moved to the $(t-1)$ tier, closer to the proximity of the user group. Moreover, the idea is made for IoT environments, which can be mobile or not. However, our work depends on a much more mobile and unstable communication environment. The work also combines reactive and proactive caching. However, for our experiment, we are only interested in comparing the results based on forming these multiple tiers with user groups' requests.

7.2.5 Results

For each run, our evaluation compares three schemes, fog cluster-based scheme (CBS), our multi-layer design without the transient cache (Multi-WTC), and finally, our multi-layer design with the transient cache (Multi-TC). We observed each run by varying either the vehicular density or the fog cluster density of the simulation. At least ten runs have been conducted for each data point to materialize the metrics' output.

In the first case, we kept the fog density fixed by eight fog nodes and varied the vehicular density by 20 to 200 to observe the different outputs of the three schemes. When the vehicular density, V_d , is a minimum of 20, the cache hits for CBS, Multi-WTC, and Multi-TC are observed as 27.3%, 25.03%, and 24.36%, respectively. As the density increases, the cache hits also increase for all three of them. However, with transient cache, the cache hit increases significantly by 44.1% for $V_d = 50$ and 49.85% for $V_d = 200$. Figure 7.2 depicts the result analyses. In Figure 7.3, for latency observation and when $V_d = 20$, the average latency was 93.51 ms, 91.531 ms, and 92.75 ms for CBS, Multi-WTC, and Multi-TC schemes, respectively. Therefore, we can infer that, due to less density of the vehicles, the content distribution of the transient cache was not balanced. However, while the density increases, the cache hit increases significantly and reduces the latency. Moreover, vehicles have limited storage capacity, making it challenging to effectively utilize the transient distributed cache for vehicle-to-vehicle (V2V) communication. This limits the content stored locally and can lead to difficulty accessing and utilizing the cache effectively. As a result, it becomes difficult to utilize the transient cache management technique for V2V-only communication at a lesser density of vehicles.

Figure 7.4 shows the results for fog density across the map for the simulation.

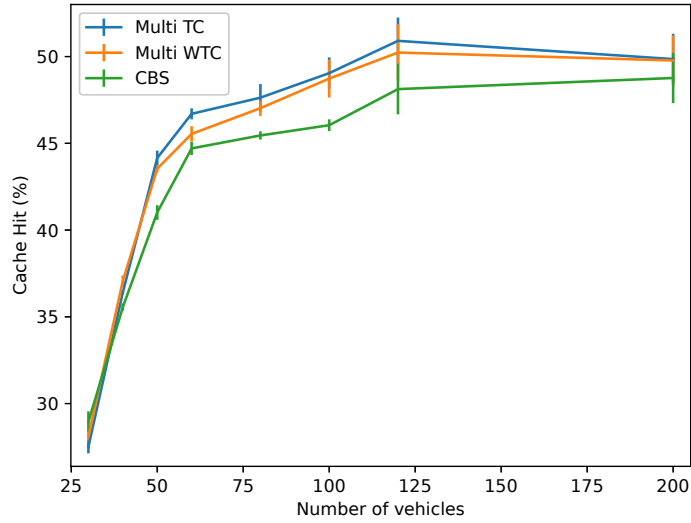


Figure 7.2: Cache Hits (%) vs Vehicular Density (8 Fog controllers).

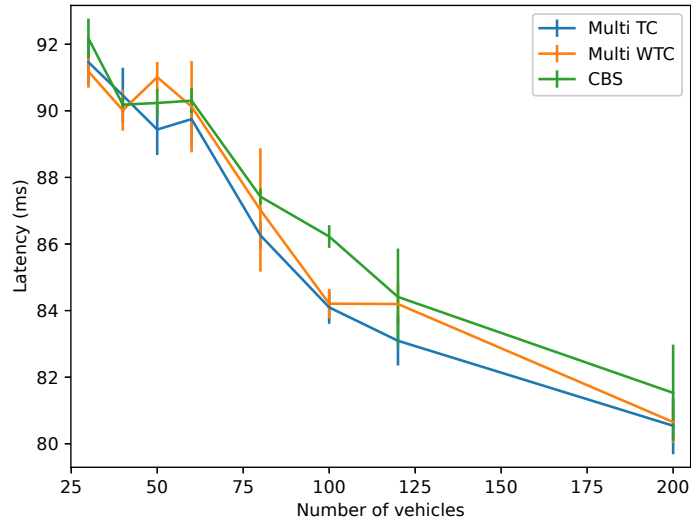


Figure 7.3: Latency (ms) vs Vehicular Density (8 Fog controllers).

For this analysis, the number of vehicles is fixed by 40. We observed the results by varying fog density, $F_d = 2, 5, 8, 10, 15$. It is seen that when $F_d = 5$, the cache hits resulted in 43.1%, 42.56%, and 41.97% for CBS, Multi-WTC, and Multi-TC schemes, respectively. This result stems from the fact that it is difficult to harvest all the benefits of a multi-layer scheme with only five fog nodes. Therefore, the cache hits' results are similar for all of them; our proposed model with transient cache performs better. When we increase the density of the fog, $F_d = 10$, the cache hits also increase to 46.1% for our model with transient cache, which is the highest among the observed schemes. With more fog nodes, $F_d = 15$, our model with transient cache performs better (46.86%) than Multi-WTC and CBS.

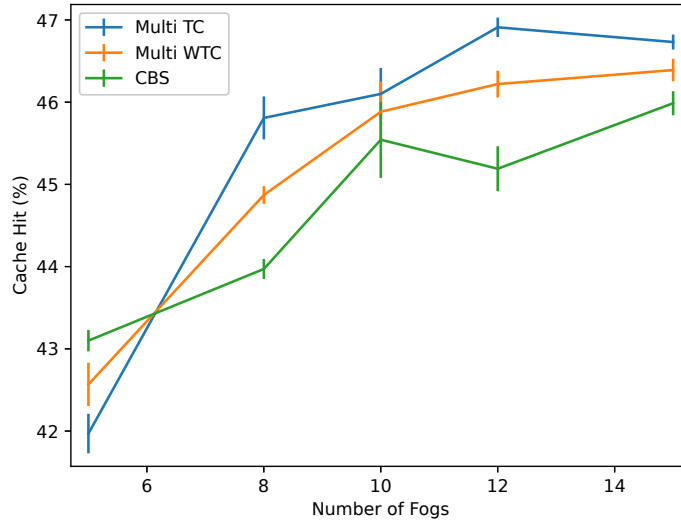


Figure 7.4: Cache Hits (%) vs Fog Density (40 vehicles).

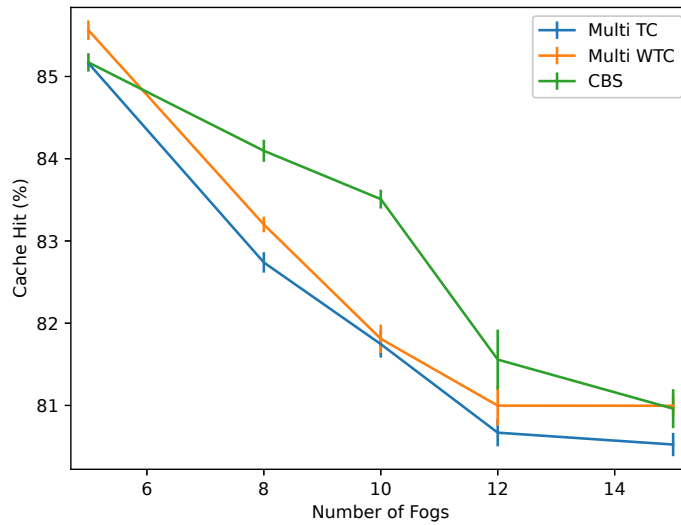


Figure 7.5: Latency (ms) vs Fog Density (40 vehicles).

In Figure 7.5, we show the analyses of varying Fog density against the average latency of the models. In this regard, $F_d = 2$ results in 85.82 ms, 85.69 ms, and 85.63 ms for our proposed model with transient cache, Multi-WTC, and CBS. However, it is seen that for $F_d = 5$, our Multi-TC shows a similar result (85.16ms) to Multi-WTC (85.56ms) and CBS (85.17ms). The distribution of the fog controller nodes can also be responsible for this oscillation - the position of the communication point of attachment impacts access directly. Thus, different placement of RSUs and a higher sampling size (number of runs) can provide smoother results - fewer oscillations. It is worth noting that the latency significantly improves while having denser Fog availability. Our model allows the system to have multiple layers with a transient

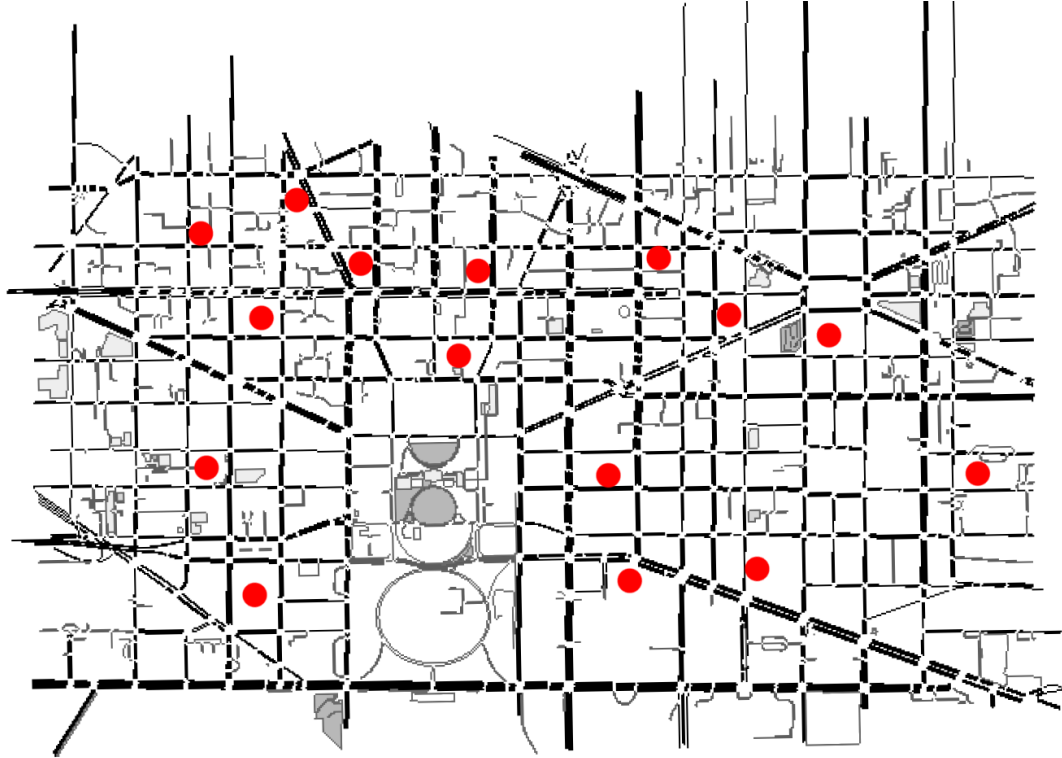


Figure 7.6: Fog node position for simulation: Washington DC, USA.

cache and more fog nodes available. It also makes the cache closer to the delivery nodes. Therefore, the latency reduces while having a transient cache closer to the delivery Fog nodes. Also, $F_d = 10$, results in a latency of 81.74 ms for our model with transient caches. Without transient caches, it shows 81.81 ms, and CBS results in 83.50 ms. Therefore, it is depicted that we can get an almost 4 ms decrease in the latency with fog density from 2 to 10 with our multi-layer approach with transient caches.

7.3 Evaluation of Popular Content Caching

In this section, we have evaluated popular content distribution with our multi-level transient caching along with zone formulation. The evaluation guides the impact of the distribution of popular content on our transient cache structure. The experiments ran through multiple iterations in an urban simulation map.

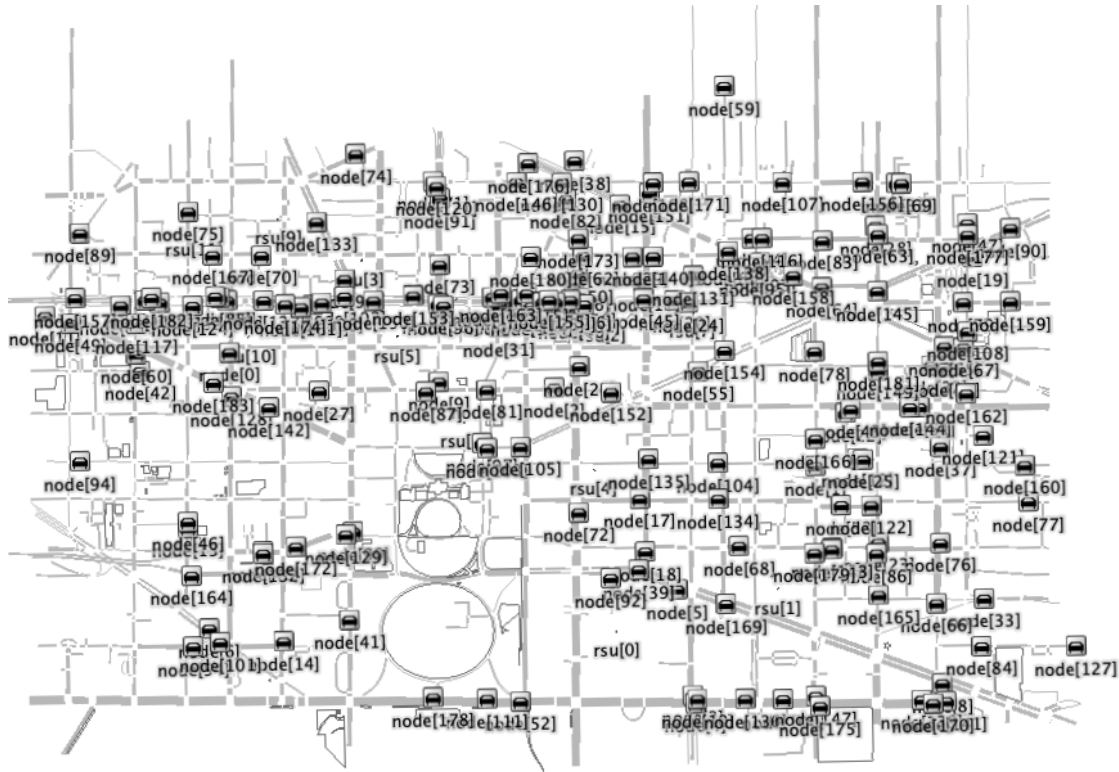


Figure 7.7: High density mobile simulation with 200 vehicles.

7.3.1 Simulation Scenario

In this evaluation, the simulation scenario is the urban area of Washington DC, USA, as shown in Figure 7.6, where the area map size is $3500 \times 3000m^2$. The area consists of complex junctions, edges, and obstacles, generating a realistic simulation environment. The red dots indicate the position of Fog nodes. In this simulation, the scenario will be much more dense and diverse with the vehicular nodes, as shown in Figure 7.7.

7.3.2 Simulation Parameters

The simulation time for our experiment has been set to 1200s due to increased iterations of the analysis to 30. The experiment is conducted in a high-density mobile scenario of 1000 vehicles and 10-20 fog nodes. The number of contents distributed among the entire network is 5×10^5 . Using Zipf's distribution law, we can distribute the contents cached at a specific frequency level. Zipf's law is a widely acknowledged popularity measurement law in the scientific community [6]. Thus the content pop-

Table 7.2: Simulation parameter settings.

Parameter	Value Range
Simulation time	1200s
Washington DC	3500 x 3000m ²
Vehicle density	20 – 1000
Vehicle Speed	0 – 50km/h
Fog Controller density	10 – 20
Node Communication range	300m
Transmission power	20mW
Number of iterations of experiment	30
Number of contents	5 x 10 ⁵
Content popularity Index	0.25 – 1
Fog Node Caching Capacity (FCC)	(1 – 5)%

ularity parameter has been set to $\alpha = (0.25 \text{ to } 1)$. The higher the α is, the more frequent the content will be preferred. Network caching capacity defines the capacity of caching of a fog node. In our experiment, we set the content capacity of the fog nodes to be constant. In terms of caching capacity, it will be the ratio of total network capacity over the number of contents generated. We have experimented with the network content capacity by setting it up as (1-5)%. Table 7.2 summarizes our simulation parameters.

7.3.3 Performance Metrics

Cache hits and latency are standardized metrics along with the cache distribution and network caching capacity parameters. Here, we have added another metric to solidify our judgment toward our experiment.

Content Hop Ratio. Content hop ratio is used to measure the hop distance of the cache placement on a server node from the requesting node, such as vehicles. In a network, a "hop" refers to a single step in forwarding a packet from one node to another. The content hop count ratio is the ratio of the number of hops a packet takes to reach the cache hit. A packet taking the maximum hops will indicate a full path stretch to reach the cache. Therefore, higher content hop ratio indicates that the packet is taking a longer path to find the cache. A low content hop ratio value indicates that the algorithm is efficient and that the network is well-connected, while a high value indicates that the algorithm is inefficient and that the network is poorly connected and the cache is not placed close enough. If in the network, max hop has

been set to Hop_{max} and it takes Hop_{org} hops to reach the target, then Content Hop Ratio (HopR) would be,

$$HopR = \frac{Hop_{org} * 100}{Hop_{max}} \quad (7.4)$$

We need to measure the average content hop ratio to testament the overall experiment, where R is the number of measured requests.

$$HopR_{avg} = \frac{\sum HopR}{R} \quad (7.5)$$

7.3.4 Protocols

For this experiment we compared three protocols through the simulations:

- **TCache-Zone.** Our proposed protocol uses a multi-layer architecture with transient caching adding locality based popular content distribution.
- **TCache / Multi-TC.** Our proposed protocol that uses a multi-layer architecture and implements transient caching across the Fogs in the urban center.
- **CBS.** This fog cluster-based caching architecture represents a recent related work [16] described in the previous sections and also used for previous comparisons.

7.3.5 Results

Our analysis was conducted by varying the content's fog node caching capacity and popularity. Each point has been simulated at least thirty times to prepare the output. Three protocols, such as transient cache with zone popularity (TCache-Zone), multi-layer transient caching (TCache), and fog Cluster-Based Scheme (CBS) are compared.

For varying content popularity and content cache capacity, we needed to keep fog density and vehicular density fixed. For this experiment, we have set vehicular density, $V_d = 200$, and fog density, $F_d = 10$. We also kept caching capacity at a fixed point of 5%.

For our first case, we have varied the content popularity to see the performance related to cache hits, latency, and content hop ratio. Content popularity, α , is varied by 0.25, 0.40, 0.60, 0.80, 1. The higher the α value, the more frequently the content is being requested. When the content popularity, α , is 0.25, the cache hits for CBS, TCache, and TCache-Zone are found to be 23.98%, 23.89%, and 27.13%, respectively. The fewer concentration of contents results in a low number of cache hits. Still, TCache-Zone has a higher percentage of scoring better due to distributed cache among

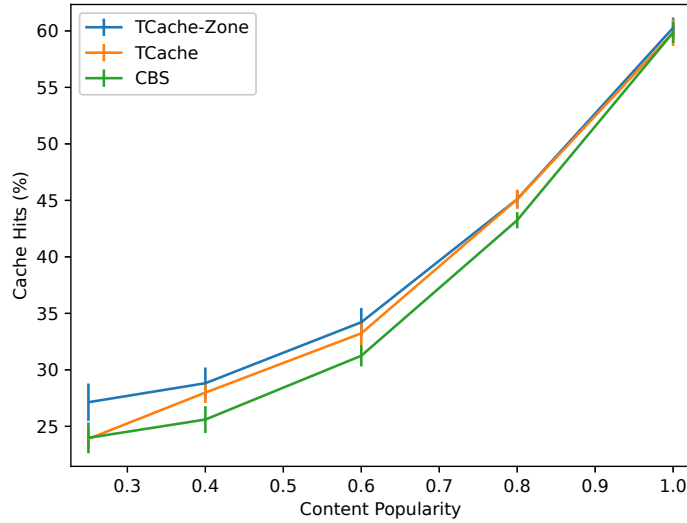


Figure 7.8: Cache Hits (%) vs Content Popularity (α) ($CC = 5\%$).

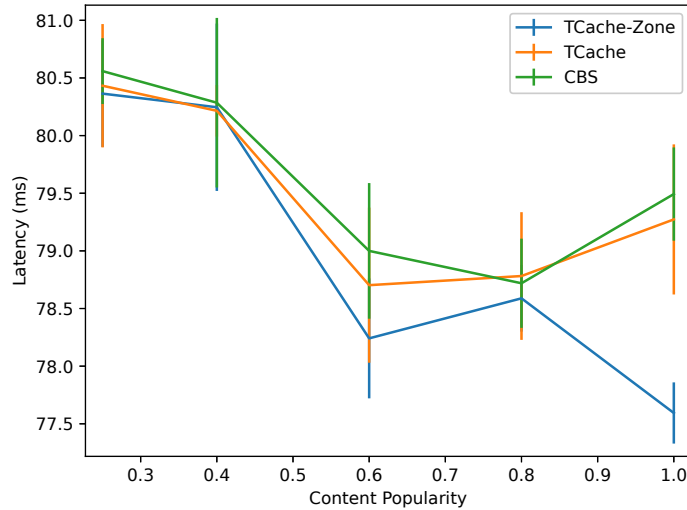


Figure 7.9: Latency (ms) vs Content Popularity (α) ($CC = 5\%$).

the fog nodes generating better-preferred contents in the cache. As content popularity increases, the cache hits for all three protocols also increase. Nevertheless, while having the most concentrated content popularity of $\alpha = 1$, TCache-Zone performs better with the cache hit of 60.2%. Figure 7.8 depicts the analysis.

In our content popularity against latency (ms) analysis, we can see that the increased density of the popularity index results in lower latency. In Figure 7.9, it is seen that while $\alpha = 0.25$, the Latency for TCache-Zone, TCache, and CBS are 80.36 ms, 80.43 ms, and 80.55 ms, respectively. However, as α increases, the time to retrieve content from the cache also reduces. At the same time, we have the highest preference for content popularity of $\alpha = 1$, results in 3 ms less time to retrieve content

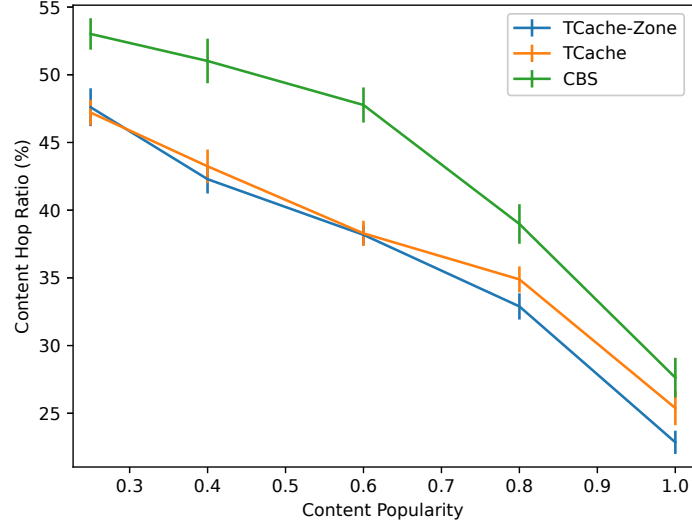


Figure 7.10: Content Hop Ratio (%) vs Content Popularity (α) ($CC = 5\%$).

from the cache.

The average content hop ratio (%) denotes how close the cache is being placed to the request originated. The lower value of the content hop ratio indicates improved cache performance. Low popularity index results lower concentration of useful content for the vehicles. Therefore, it is assumable that a minimum value of $\alpha = 0.25$ will result in a high content hop ratio. Therefore, TCache-Zone, TCache, and CBS are 47.60%, 47.20%, and 53.0161%, respectively. TCache-Zone and TCache perform quite similarly as both protocols follow the same multi-layer strategy, which aims to place the cache from top to bottom. Hence, for a higher α value TCache-Zone inherits the most prominent content to be utilized near the delivery path. As a result, we can see from Figure 7.10 that, for $\alpha = 1$, the output for TCache-Zone = 22.85%, which is around 5% less than of CBS = 27.62%.

In the following case, we have varied the caching capacity to see the cache hits, latency, and content hop ratio performance. For this analysis, we have set vehicular density, $V_d = 1000$, and fog density, $F_d = 10$. Caching capacity, CC is varied by 1%, 2%, 3%, 4%, 5%. The higher the CC value, the more useful content can be placed in the cache and provides a caching algorithm to manage non-frequent content eviction. From Figure 7.11, we can observe that when the CC is 1%, the cache hits for TCache-Zone, TCache, and CBS are found to be 28.28%, 28.09%, 24.86%, respectively. The higher concentration of contents results in a low number of cache hits. Still, TCache-Zone has a higher percentage of scoring better due to distributed cache among the fog nodes generating better-preferred contents in the cache. As the content popularity

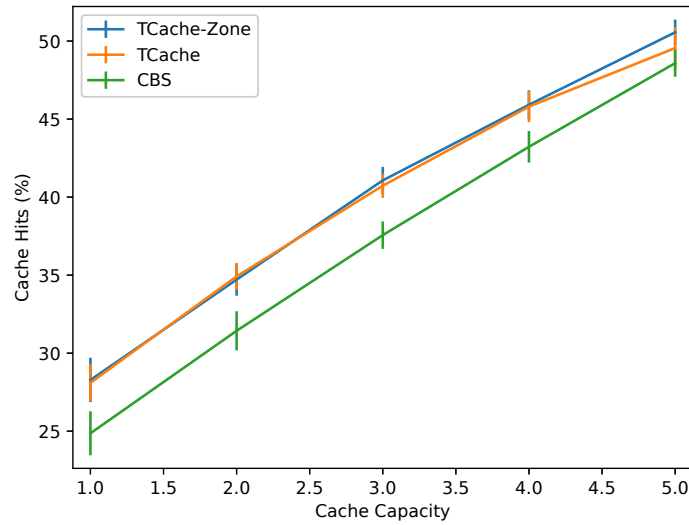


Figure 7.11: Cache Hits (%) vs Cache Capacity (CC) ($\alpha = 0.6$).

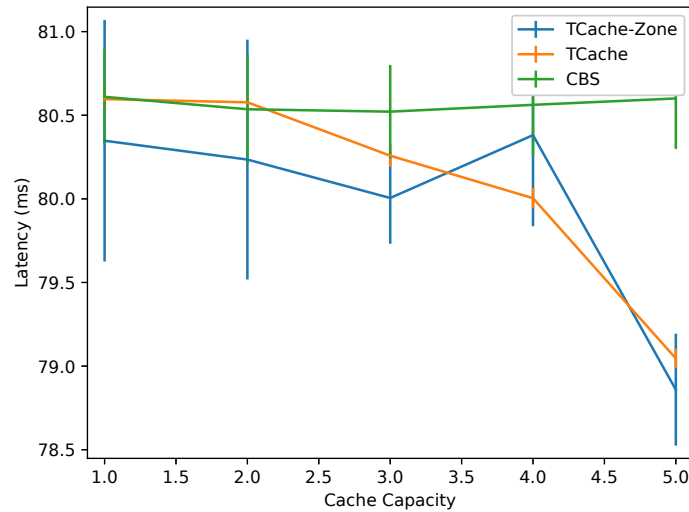


Figure 7.12: Latency (ms) vs Cache Capacity (CC) ($\alpha = 0.6$).

increases, the cache hits also increase for all three protocols. Nevertheless, while having the most concentrated content popularity of $\alpha = 1$, TCache-Zone performs the most better with the cache hit of 60.2%. For this observation, the content popularity index has been kept fixed to $\alpha = 0.6$.

We observed some circumstances in Figure 7.12 of latency evolution against cache capacity. The number of contents to be placed in the cache depends on the caching capacity. The less cache capacity means fewer contents can be placed in the cache. Therefore, the content fetch takes longer and results in high latency. From $CC = 1.0\%$ observation point, the latency is between 80.34 ms, 80.59 ms, and 80.61 ms, respectively, for TCache-Zone, TCache, and CBS. At one point, while $CC = 4.0\%$,

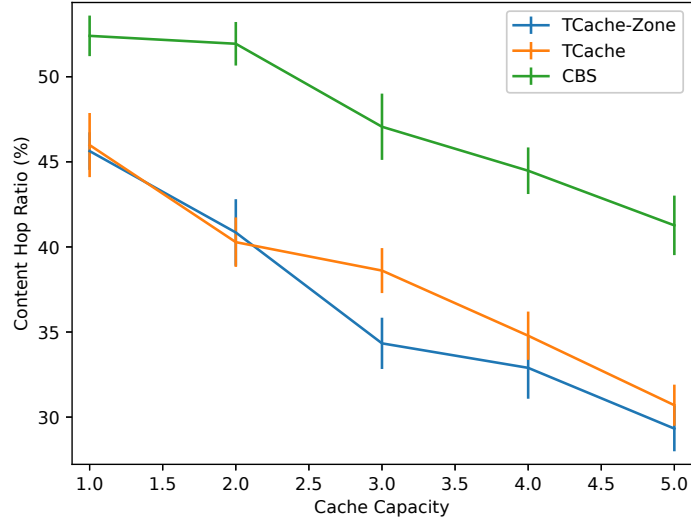


Figure 7.13: Content Hop Ratio (%) vs Cache Capacity (CC) ($\alpha = 0.6$).

TCache-Zone achieves a higher latency of 80.38 ms, 80.004 ms than the original TCache. In our system, the vehicles have high mobility. The point vehicle requested the content might not be on the same point a few seconds after. Therefore, the delivery path can be different while measuring the latency. Apart from this scenario, the latency significantly reduces while the capacity increases. At the most cache capacity of 5%, the TCache-Zone has approximately 4 ms of lesser latency than CBS.

Figure 7.13 depicts the average content hop ratio (%) against the network caching capacity. Lower caching capacity results in lower storage for cache contents, while the cache eviction must be high. As a result, the content hop ratio is significant while $CC = 1\%$, TCache-Zone = 45.6326%, TCache = 45.98 %, and CBS = 52.40%. Still, the transient cache performed better as a multi-level cache in place in order to support the vehicles. Hence, similar to content popularity and content hop ratio evaluation, we can see that TCache-Zone and TCache perform pretty similarly as both protocols follow the same multi-layer strategy aiming to place cache from top to bottom. Although, for higher CC value TCache-Zone $CC = 5\%$, the output for TCache-Zone = 29.32%, around 10% less than of CBS = 40.26%.

7.4 Remarks

In summary, we showed that using a multi-layer transient cache proactively increases cache hits and reduces latency. The strategy has improved performance by varying vehicular and fog density. Our multi-layer transient caching has about 2% improve-

ment in cache hits and 1ms reduced latency against different vehicle numbers and almost (0.9 - 1.6)% and (0.47 - 1.35)ms against varying fog numbers. However, the strategy will only be effective at a certain point if it can identify which content is more useful or popular. Therefore, the zone and frequency meter calculated the worthiness of the content and noted it as popular content. This strategy further improved cache hits, reduced latency, and brought caches closer to the hop count of their particular interests. Our strategy has improved cache hits and reduced latency by about (0.5 - 1.8)% and (0.7 - 1.6)ms on varied content popularity. In terms of caching capacity, it has improved cache hits by almost 2.2% and reduced latency by 1.7ms. Furthermore, our model brings the cache closer to the vehicle proximity, which shows a reduced average hop count ratio of almost 10% and 6% against different cache capacities and content popularity parameters. The scheme also includes a data replication mechanism, which allows for multiple copies of the same data to be stored in different fog nodes, ensuring the availability of the data, even in case of node failures.

Chapter 8

Conclusion

Caching is critical in enhancing the performance of systems that handle heavy traffic. However, due to the mobility of vehicles, the demand for dynamic content, and the accessibility of the resources for processing, serious challenges exist in the vehicular cloud computing environment. Therefore, this study looked at the prospect of temporarily storing content and how the mobility of the vehicles might influence the popularity of that content.

8.1 Summary

The system incorporates a multi-level caching structure to make content available closer to the content interests. The multi-level cache is structured mainly with three layers: primary, fog, and edge. The fog layer can also create sub-layers in its perimeter. The edge layer consists of vehicles generating requests to reach the fog layer. The content is being searched throughout the fog and sub-layers, forwarding to the primary layer. Additional content is being added with the request in order to place it in the transient cache.

Our study investigated the local interests of contents and prospects for the transient cache to store more worthwhile content. In addition to the multi-level design, zone allocation in caching has also been used to find the region-based interests in the network. While storing the content in the cache, it keeps track of how frequently a particular content is being requested, measures the age time limit in the cache, and notes the zone that it originated from. Therefore the system is able to fetch top contents to serve with the responses.

The performance analysis compared our proposed caching architecture and model with a previous design that targets cooperative caching to support ICN. We have

also observed the performance of transient caches with popularity in our proposed system. Results are shown based on the performance metrics of cache hit ratio, request latency, and average content hop ratio considering vehicular density, fog density, content popularity, and caching capacity as the performance parameters along with the aforementioned simulation configuration. The overall results demonstrated that the proposed model with transient caches using zoning and popularity indexing generates a better cache hit ratio and latency and generates efficient cache placement with an average hop count ratio.

The system's limitations include the additional load due to the implementation of the transient cache, which may impact the system's performance under a large number of requests. Additionally, the optimized fog selection process and delivery path selection may result in an optimal distribution of resources, potentially leading to better latency reduction. Furthermore, the service and user storage permission may restrict access to certain features, hindering the system's overall features. These limitations highlight the need for ongoing optimization and improvement to ensure a high-quality user experience.

8.2 Future Work Directions

- **5G Network.** 5G is the fifth generation of mobile networks, which offers faster data rates, lower latency, and increased capacity compared to previous generations of mobile networks. This makes it well-suited for supporting the high-bandwidth and low-latency requirements of vehicular networks. Therefore cache decision mechanisms over 5G and RSU communication arise as a compelling solution.
- **Cache Delivery.** Path prediction of cache delivery is always challenging in vehicular networks due to the dynamic routing of the vehicles. A more efficient cache delivery strategy stressed in high mobility also can be evaluated.
- **Learning Model.** With the help of intelligent learning of vehicles, machine learning algorithms and predictive models with personalized caching strategies in local regions may further shape useful content caching.

Bibliography

- [1] Md Maruf Ahamed and Saleh Faruque. 5g network coverage planning and analysis of the deployment challenges. *MDPI Sensors*, 21(19):6608, 2021.
- [2] Marica Amadeo, Claudia Campolo, and Antonella Molinaro. Priority-based content delivery in the internet of vehicles through named data networking. *MDPI Journal of Sensor and Actuator Networks*, 5(4):17, 2016.
- [3] L. A. Belady. A study of replacement algorithms for a virtual-storage computer. *IBM Systems Journal*, 5(2):78–101, 1966.
- [4] Muhammad Bilal and Shin-Gak Kang. Time aware least recent used (tlru) cache management policy in icn. In *Proceedings of the IEEE International Conference on Advanced Communication Technology*, pages 528–532, 2014.
- [5] Abdelwahab Boualouache, Ridha Soua, and Thomas Engel. Toward an sdn-based data collection scheme for vehicular fog computing. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 1–6, 2020.
- [6] L. Breslau, Pei Cao, Li Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: evidence and implications. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM) - Annual Joint Conference of the IEEE Computer and Communications Societies (The Future is Now - Cat. No.99CH36320)*, volume 1, pages 126–134, 1999.
- [7] Chen Chen, Cong Wang, Tie Qiu, Mohammed Atiquzzaman, and Dapeng Oliver Wu. Caching in vehicular named data networking: Architecture, schemes and future directions. *IEEE Communications Surveys & Tutorials*, 22(4):2378–2407, 2020.
- [8] Shanzhi Chen, Jinling Hu, Yan Shi, Ying Peng, Jiayi Fang, Rui Zhao, and Li Zhao. Vehicle-to-everything (v2x) services supported by lte-based systems and 5g. *IEEE Communications Standards Magazine*, 1(2):70–76, 2017.

- [9] Ashit Kumar Dutta, Mohamed Elhoseny, Vandna Dahiya, and K. Shankar. An efficient hierarchical clustering protocol for multihop internet of vehicles communication. *Wiley Transactions on Emerging Telecommunications Technologies*, 31(5):e3690, 2020.
- [10] Samira El Madani, Saad Motahhir, and El Ghzizal Abdelaziz. Internet of vehicles: concept, process, security aspects and solutions. *Multimedia Tools and Applications*, 81:16563–16587, 2022.
- [11] Rim Gasmi and Makhlouf Aliouat. Vehicular ad hoc networks versus internet of vehicles - a comparative view. In *Proceedings of the IEEE International Conference on Networking and Advanced Systems (ICNAS)*, pages 1–6, 2019.
- [12] Christoph Sommer. Reinhard German. and Falko Dressler. Bidirectionally coupled network and road traffic simulation for improved ivc analysis. *IEEE Transactions on Mobile Computing*, 10(1):3–15, 2011.
- [13] Marco Giordani, Andrea Zanella, Takamasa Higuchi, Onur Altintas, and Michele Zorzi. *Springer Emerging Trends in Vehicular Communication Networks*, pages 37–57. Springer Singapore, 2018.
- [14] Sohan Gyawali, Shengjie Xu, Yi Qian, and Rose Qingyang Hu. Challenges and solutions for cellular based v2x communications. *IEEE Communications Surveys Tutorials*, 23(1):222–255, 2021.
- [15] K M Nafiul Hassan and Robson E. De Grande. Mobility-based multi-layered caching and data distribution in vehicular fog computing. In *Proceedings of the IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, pages 164–167, 2022.
- [16] Yining Hua, Lin Guan, and Konstantinos Kyriakopoulos. A fog caching scheme enabled by icn for iot environments. *Elsevier Future Generation Computer Systems*, 111:82–95, 2020.
- [17] Xiaoge Huang, Ke Xu, Qianbin Chen, and Jie Zhang. Delay-aware caching in internet-of-vehicles networks. *IEEE Internet of Things Journal*, 8(13):10911–10921, 2021.
- [18] Rodolfo I. Meneguette, Robson E. De Grande, and Antonio A. F. Loureiro. Intelligent transportation systems. In *Intelligent Transport System in Smart*

- Cities: Aspects and Challenges of Vehicular Networks and Cloud*, pages 1–21. Springer International Publishing, Cham, 1 edition, 2018.
- [19] Muhammad Awais Javed and Sherali Zeadally. Ai-empowered content caching in vehicular edge computing: Opportunities and challenges. *IEEE Network*, 35(3):109–115, 2021.
- [20] Qiang Li, Yuanmei Zhang, Yingyu Li, Yong Xiao, and Xiaohu Ge. Capacity-aware edge caching in fog computing networks. *IEEE Transactions on Vehicular Technology*, 69(8):9244–9248, 2020.
- [21] Kedong Liu, Yanwei Liu, Jinxia Liu, and Antonios Argyriou. Tile caching for scalable vr video streaming over 5g mobile networks. *Elsevier Journal of Visual Communication and Image Representation*, 79:103210, 2021.
- [22] L. Liu, C. Chen, Qingqi Pei, Sabita Maharjan, and Yu lei Zhang. Vehicular edge computing and networking: A survey. *Springer Mobile Networks and Applications*, 26:1145–1168, 2021.
- [23] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wiessner. Microscopic traffic simulation using sumo. In *Proceedings of the International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582, 2018.
- [24] Junchao Ma, Jiahuan Wang, Gang Liu, and Pingzhi Fan. Low latency caching placement policy for cloud-based vanet with both vehicle caches and rsu caches. In *Proceedings of the IEEE Global Communications Conference Workshop (GLOBECOM)*, pages 1–6, 2017.
- [25] Xiao Ma, Ao Zhou, Shan Zhang, and Shangguang Wang. Cooperative service caching and workload scheduling in mobile edge computing. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, pages 2076–2085, 2020.
- [26] Zhaolong Ning, Kaiyuan Zhang, Xiaojie Wang, Mohammad S. Obaidat, Lei Guo, Xiping Hu, Bin Hu, Yi Guo, Balqies Sadoun, and Ricky Y. K. Kwok. Joint computing and caching in 5g-envisioned internet of vehicles: A deep reinforcement learning-based traffic control system. *IEEE Transactions on Intelligent Transportation Systems*, 22(8):5201–5212, 2021.

- [27] Ying Sai, Dong zhu Fan, and Meng yang Fan. Cooperative and efficient content caching and distribution mechanism in 5g network. *Elsevier Journal of Computer Communications*, 161:183–190, 2020.
- [28] Sukhmani Sukhmani, Mohammad Sadeghi, Melike Erol-Kantarci, and Abdulmoteleb El Saddik. Edge caching and computing in 5g for mobile ar/vr and tactile internet. *IEEE MultiMedia*, 26(1):21–30, 2019.
- [29] Rehmat Ullah, Muhammad Atif Ur Rehman, Muhammad Naeem, Byung-Seo Kim, and Spyridon Mastorakis. Icn with edge for 5g: Exploiting in-network caching in icn-based edge computing for 5g networks. *Elsevier Future Generation Computer Systems*, 111:159–174, 2020.
- [30] Andras Varga and Rudolf Hornig. An overview of the omnet++ simulation environment. In *Proceedings of the International Conference on Simulation Tools and Techniques for Communications, Networks and Systems and Workshops (Simu-tools)*, page 1–10, 2008.
- [31] Cong Wang, Chen Chen, Qingqi Pei, and Licheng Wang. Idds: An icn based data dissemination scheme for vehicular networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 1–7, 2020.
- [32] Ruyan Wang, Zunwei Kan, Yaping Cui, Dapeng Wu, and Yan Zhen. Cooperative caching strategy with content request prediction in internet of vehicles. *IEEE Internet of Things Journal*, 8(11):8964–8975, 2021.
- [33] Yu Xiao and Chao Zhu. Vehicular fog computing: Vision and challenges. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 6–9, 2017.
- [34] Xiaolong Xu, Zijie Fang, Jie Zhang, Qiang He, Dongxiao Yu, Lianyong Qi, and Wanchun Dou. Edge content caching with deep spatiotemporal residual network for iov in smart city. *ACM Transactions on Sensor Networks*, 17(3):1–33, 2021.
- [35] Zichuan Xu, Shengnan Wang, Shipei Liu, Haipeng Dai, Qiufen Xia, Weifa Liang, and Guowei Wu. Learning for exception: Dynamic service caching in 5g-enabled mecs with bursty user demands. In *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 1079–1089, 2020.
- [36] Rahul Yadav, Weizhe Zhang, Omprakash Kaiwartya, Houbing Song, and Shui Yu. Energy-latency tradeoff for dynamic computation offloading in vehicular fog

- computing. *IEEE Transactions on Vehicular Technology*, 69(12):14198–14211, 2020.
- [37] Ruihang Yang and Songtao Guo. A mobile edge caching strategy for video grouping in vehicular networks. In *Proceedings of the IEEE International Conference on Advanced Computational Intelligence (ICACI)*, pages 40–45, 2021.
- [38] Junhui Zhao, Xiaoke Sun, Qiuping Li, and Xiaoting Ma. Edge caching and computation management for real-time internet of vehicles: An online and distributed approach. *IEEE Transactions on Intelligent Transportation Systems*, 22(4):2183–2197, 2021.