Received February 5, 2020, accepted February 26, 2020, date of publication March 19, 2020, date of current version April 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2981928

Towards Efficient NDN Framework for Connected Vehicle Applications

NING YANG[1], (Student Member, IEEE), KANG CHEN[1], (Member, IEEE), AND YAOQING LIU[2], (Member, IEEE)

[1]Department of Electrical and Computer Engineering, Southern Illinois University, Carbondale, IL 62901, USA
[2]School of Computer Sciences and Engineering, Fairleigh Dickinson University, Teaneck, NJ 07666, USA

Corresponding author: Kang Chen (kchen@siu.edu)

This work was supported in part by the Startup Fund and in part by the Energy Boost Seed Grant from the Advanced Coal and Energy Research Center at Southern Illinois University Carbondale.

ABSTRACT Named Data Networking (NDN) recently arises as a promising technology to support connected vehicle (CV) applications due to the match between their characteristics. However, the fast mobility and the vast number of vehicles raise great challenges in designing a scalable and efficient NDN network for CV applications. Therefore, in this paper, we develop an NDN based CV application framework that handles the challenge through innovations in two aspects. First, we propose a hierarchical hyperbolic NDN backbone architecture (H2NDN). H2NDN exploits the location dependency of CV applications to develop a hierarchical router topology and a hierarchical data/interest namespace. As a result, efficient and scalable data retrieval can be achieved by only configuring static forwarding information base (FIB) on NDN routers. To avoid overloading high-level routers, H2NDN integrates hyperbolic routing into the hierarchical architecture through carefully designed hyperbolic planes. Second, on top of the H2NDN architecture, we further model the optimal data caching problem. Based on the modeling, we propose a distributed adaptive caching strategy that can greatly improve the efficiency of the H2NDN backbone in supporting CV applications. Extensive ndnSIM based experiments with real traffic data in a city prove the efficiency and scalability of the proposed NDN application framework.

INDEX TERMS Connected vehicle, named data networking, connected vehicle applications.

I. INTRODUCTION

Connected vehicle (CV) technology has been developed recently to provide wireless networking connectivity to vehicles, thus interconnecting them with each other as well as with infrastructures [1], [2]. With such connectivity, many beneficial CV applications could be developed to tackle some of the biggest challenges in the surface transportation industry such as safety, mobility efficiency, and environmental protection [3]. Exemplary CV applications include collision warning, cooperative adaptive cruise control, real-time route planning, eco-speed harmonization, etc. In recognition of its potential, the U.S. Department of Transportation has already launched a pilot program to deploy, test, and operationalize cutting-edge CV technologies and applications [4].

In this paper, we focus on CV applications that are designed to exploit location-related information for social/human benefits such as route planning. Our careful examination finds that the current IP-based Internet architecture faces some challenges in supporting such CV applications. The major reason is that these CV applications are location dependent rather than identity dependent. They mainly rely on information related to certain geographic locations rather than specific vehicles. Thus the IP addressability becomes a burden rather than a blessing for CV applications (please refer to Section III-C for detailed drawbacks of the IP architecture in supporting CV applications). Meanwhile, on the other hand, named data networking (NDN) [5] matches the aforementioned needs of CV applications by distributing data through names instead of IP addresses. Moreover, its native support of in-network caching, multi-path routing, and adaptive forwarding can further improve CVs' data retrieval efficiency. Consequently, NDN provides an effective communication model for CV applications.

To this end, we propose an NDN-based CV application framework to enable CV applications to retrieve data from

The associate editor coordinating the review of this manuscript and approving it for publication was Junhui Zhao.

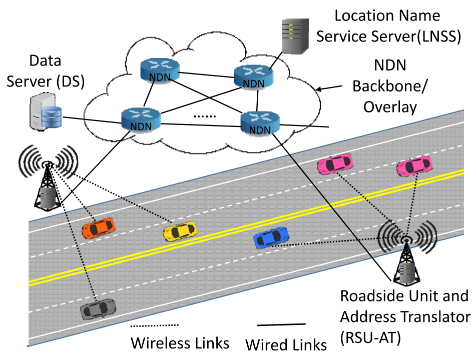


FIGURE 1. **Overview of the NDN based CV application framework.**

data servers (DSs) efficiently. Our Figure 1 illustrates this framework. It is not hard to find that, due to the fast mobility and the vast number of vehicles, the NDN framework has to be able to span a large area and serve a high volume of queries efficiently (i.e., demanding a high scalability and a high efficiency). However, such a need has not been comprehensively studied in current literature. Thus, in this paper, we develop solutions to handle this challenge in two aspects. We first propose a novel hierarchical hyperbolic NDN backbone architecture (H2NDN) that deeply integrates the characteristics of CV applications and NDN. We then propose a distributed adaptive algorithm that can greatly improve the efficiency of caches on NDN routers.

The H2NDN backbone develops a hierarchical router topology and data/interest namespace by using the hierarchical nature of geographic locations (e.g., state→city→road). This allows the backbone to forward Interest packets towards their target DSs (i.e., DSs that store the requested data) by just installing static FIB entries on NDN routers (i.e., by comparing the location of the target DS with the location of the current router). However, purely following the hierarchical topology to forward Interest packets (i.e., hierarchical routing) can easily overload high-level routers in the topology. We solve this problem by offloading the traffic to high-level routers through carefully designed hyperbolic routing. A novel scheme is developed switch between hierarchical routing and hyperbolic routing seamlessly, thus ensuring the correctness and efficiency of data retrieval. As a result, the hierarchical architecture's advantages are kept while the challenge on scalability is greatly alleviated. The H2NDN backbone is introduced in Section IV.

Moreover, since vehicles may repetitively request the same data, we further exploit the built-in caching capability of NDN routers to improve the data retrieval efficiency of CV applications. Particularly, we study how to optimize the creation of data caches on routers (which usually own a limited cache space) in the H2NDN architecture. We first find that global optimization is not practical due to complexity and scalability issues. We then propose a distributed algorithm that lets data compete with each other directly on routers based on their potentials in saving forwarding hops for future Interest packets. Consequently, popular data is more likely to be cached on routers close to vehicles where they can save the most hops, while caches of less-popular data are pushed towards data servers. This lowers the average number of forwarding hops needed to hit data adaptively. The detail of the caching algorithm is presented in Section V.

In summary, our major contributions include:

- We propose an NDN based CV application framework by exploiting the synergies between NDN and CV applications.
- We design a novel hierarchical hyperbolic NDN backbone architecture (H2NDN) in the application framework. H2NDN deeply integrates the characteristics of CV applications to enable scalable and efficient data retrieval from DSs.
- We propose a distributed adaptive cache placement algorithm in the H2NDN backbone to further improve its performance in supporting CV applications.
- Extensive ndnSIM-based experiments with a real traffic trace demonstrate the high performance of the CV application framework.

In the remaining of this paper, related work is discussed in Section II. The preliminaries and overview of the NDN based CV application framework are presented in Section III. Sections IV and V introduce the hierarchical hyperbolic NDN backbone architecture and the cache allocation strategy adopted in the framework, respectively. Section VI presents our evaluation results using ndnSIM. Finally, Section VII concludes the paper with remarks on future work.

II. RELATED WORK

In this section, we first introduce existing studies that employ NDN to support vehicle-to-vehicle and vehicle-to-infrastructure communication. We then present the state-of-arts of two important components of the proposed H2NDN, i.e., hyperbolic routing and data caching in NDN.

A. EMPLOY NDN TO SUPPORT VEHICLES

Employing NDN to enhance the performance of vehicular networks has attracted much attention recently. The surveys in [6], [7] comprehensively discuss the advantages of this direction (e.g., handle mobility, security, and scalability issues) as well as the open research problems and challenges (e.g., naming, forwarding, caching, and architecture).

Early researches focus on the application of NDN in the context of the vehicular ad hoc network (VANET), i.e., vehicle-to-vehicle communication. Amadeo *et al.* [8] conducted the pioneering study that shows the benefits of information-centric networking (ICN) over the traditional IP-based network architecture in VANETs. The advantages are also proved in [9], [10] through rigorous simulations and tests. The studies in [11], [12] propose to use collision avoidance and selective flooding to enhance the performance of content routing among vehicles. To handle the broadcast storm issue in vehicular NDNs, CODIE [13] limits the maximal forwarding hops of Interest/Data packets, DIFS [14] lets the packet receiver check its validity as a forwarder, and LISIC [15] prioritizes Interests by predicted link stability. Grassi *et al.* [16] designed a geolocation-based naming scheme for vehicular NDNs in suburban areas. The work in [17] exploits a tree-based Data/Interest structure to achieve efficient NDN-based traffic information dissemination among vehicles. Lin *et al.* [18] handled the topology change in VANETs by constructing reliable forwarding paths based on mobility information. MobiVNDN [19] adapts the NDN framework to handle unique issues in vehicular NDNs. Deng *et al.* [20] exploits NDN to enable efficient large file transfer among neighbor vehicles.

NDN can also be employed in roadside backbone infrastructures to enhance the data communication to/from vehicles, i.e., vehicle-to-infrastructure communication. Jiang *et al.* [21] proved that NDN can effectively reduce the delay of delivering data to vehicles through a federated simulation platform. The work in [22] exploits NDN to improve the efficiency of data retrieval from vehicles at mobility. ADePt [23] pre-fetches data for vehicles along their trajectory. The trajectory information is obtained by analyzing their past Interest and Data packets. PeRCeIVE [24] proactively facilitates the data cache at roadside units with the side information (i.e., mobility and interests) reported by vehicles. The study in [25] adopts an entropy-based proactive caching scheme to handle the uncertainty in mobility prediction. The work in [26] models the optimal content caching on RSUs in the V2X context and proposes a framework to evaluate the influence of multiple topology- and network-related parameters. Kurihara and Filali [27] proposed a hierarchical name/data structure to facilitate the NDN-based location data retrieval. Drira *et al.* [28] adapted NDN to provide the Pub/Sub service to vehicles. The work in [29] adopts NDN and a hierarchal namespace to disseminate safety information to vehicles in a publish-subscribe manner.

We see that existing studies mostly directly exploit the features of NDN (i.e., data caching and content-based routing) to facilitate the data retrieval from or dissemination to vehicles and assume a general NDN architecture. They fail to further study how to improve the architecture of the NDN backbone in providing scalable and efficient data services to CV applications. This study thus is proposed to fill this gap. The work that is most similar to ours is [30], which proposes a pure hierarchical router topology and namespace in the vehicular NDN. However, such a router topology presents limited scalability (as discussed later in Section IV-B). The paper also fails to improve the caching on NDN routers.

B. HYPERBOLIC ROUTING IN NDN

Hyperbolic routing (i.e., map routers to the hyperbolic space for packet routing) has the potential to solve the scaling limitation faced by the routing in large networks [31]–[34]. The work in [31] shows that every connected finite graph has a greedy embedding in the hyperbolic plane through which greedy geometric routing can always succeed. The studies in [32], [33] also demonstrate the connection between scale-free complex networks and hyperbolic geometries. The work in [34] proposes an approach to map the Internet to the hyperbolic space, over which the greedy forwarding achieves close-to-optimal routing efficiency.

Such an advantage naturally drives the application of hyperbolic routing in NDN (which also suffers from the scaling challenge) [35]. A number of experimental studies have been carried out to evaluate the performance of hyperbolic routing in NDN [36]–[38]. They all show that the hyperbolic routing presents a promising performance (i.e., close with the best results or that of link-state algorithms). However, the network size in these experiments is quite limited (i.e., less than 100). The work in [39] proposes to consider both betweenness centrality and content popularity when computing the hyperbolic embedding of nodes in the network. But it assumes static content popularity, which does hold in practical scenarios.

C. DATA CACHING IN NDN

NDN provides native support to cache data at routers for future Interest packets [5]. Such a feature potentially can greatly improve the efficiency of data retrieval through NDN. To this end, a lot of research efforts have been devoted to improving the cache efficiency in NDN [40]–[46]. The studies can be divided into two categories.

The first category studies the allocation of cache space to routers under constrained total available cache space and the appropriate caching location. Wang *et al.* [40] studies the optimal allocation of cache capacity to routers under the constraint of total cache space. The work in [41] studies the performance of allocating the cache space according to several graph-related centrality metrics and concludes that the degree centrality metric performs well enough. Dabirmoghaddam *et al.* [45] find that optimal caching along the forwarding path only presents a slight performance gain over caching opportunistically at the edge.

The second category works on the cache replacement on individual routers. Intuitive cache allocation strategies include First-In-First-Out (FIFO), Least-Recently-Used (LRU), and Least-Frequently-Used (LFU) [5]. It is not hard to find that those strategies could easily create duplicate caches along the forwarding path and cause a low utilization efficiency of cache spaces. The work in [46] reduces the cache duplication by caching data over the node on the forwarding path that has the largest betweenness centrality. We see that such a strategy may not fully utilize the cache space of all nodes. To handle this issue, ProbCache [42], [43] caches data probabilistically along the forwarding path, and the probability is jointly decided by the availability of cache space in the remaining routers of the path and the percentage of hops that have already been traversed. However, it fails to consider data popularity when caching data and assumes that routers own enough cache spaces. Prob-PD [44] thus integrates both potential benefit and data popularity into the caching probability. But it computes the benefit as the percentage of hops saved, which only shows the relative number of hops saved. However, for performance evaluation, it should consider the real numbers of saved hops. Therefore, both ProbCache and Prob-PD cannot optimize the total number of hop savings, which is achieved by the caching strategy in this work.

III. NDN BASED CV APPLICATION FRAMEWORK: PRELIMINARIES AND OVERVIEW

In this section, we first discuss important preliminary thoughts regarding the design principle of the proposed framework: 1) CV scenario; 2)target CV applications; 3) the rationale of adopting NDN for CV applications; and 4) the data management model. We then present the overview of the NDN based CV application framework along with an exemplary CV application. The designs of two core components of the framework are presented later in Sections IV and V.

A. CV SCENARIO

In this paper, we define CV as vehicles that own the wireless networking capability while moving on roads. Such connectivity enables CVs to achieve real-time data communication with servers on the Internet (e.g., cloud servers) and other vehicles. Depending on the type of communication end-host, CV communication can be categorized into three scenarios: vehicle-to-vehicle (V2V), vehicle-to-roadside unit (V2R), and vehicle-to-infrastructure (V2I) [47], [48].

While all CV scenarios are promising, we focus on the vehicle-to-infrastructure scenario since it enables the accessibility to the vast amount of resources available on the Internet (which can be employed to develop many powerful applications). However, developing applications in the V2I scenario faces challenges in multiple aspects including wireless link efficiency [48]–[50], computing offloading [51], [52], and network architecture and management [30], [53], [54]. In this work, we aim to design a novel NDN-based network architecture to support CV applications in the V2I scenario considering its wide impact.

B. TARGET CV APPLICATIONS

CV applications refer to those that exploit the networking connectivity on vehicles to improve various aspects of the transportation system such as safety, road surveillance and maintenance, mobility efficiency, and environmental protection [3]. According to the USDoT CV pilot program, several dozens of CV application concepts have been developed so far [4]. These applications can be divided into two categories according to the geographic area they consider. The first category of applications is developed upon information from proximity vehicles. Examples in this category include collision avoidance and cooperative adaptive cruise control. The other category of applications collects information in broad areas for beneficial services. Example applications are trip planning and traffic scheduling.

In this research, we focus on the second category of CV applications, i.e., those that rely on timely dissemination/collection of vehicle/traffic-related information in broad areas. Such information often is location dependent. We are interested in such applications because they face many challenges due to the large scale. We aim to build an NDN framework in supporting those applications.

We are aware that there are many applications that are not location-dependent such as entertainment video and phone call. We claim that these applications can be adapted to work in the proposed NDN framework. Specifically, the Location Name Service Server (LNSS) proposed in this paper, i.e., shown later in Figure 1, can provide the location of the server that stores the requested data or connects to the receiver of the phone call. Then, the corresponding Interest packets can take the server location as the prefix, which allows the NDN framework to forward them to the corresponding data server. We leave the details of such adaptations to future work.

C. WHY NDN?

Intuitively, building CV applications over the current Internet seems to be an easy-to-implement option due to the mature technical foundations. However, the Internet architecture assumes a connected environment builds upon the IP addressability, in which vehicles must remain connected with application servers or other vehicles. This raises some problems. First, the connection-oriented architecture makes it hard to scale to a large volume of vehicles. Second, vehicle mobility causes frequent handover among wireless access points (which also leads to the change of IP address), which challenges the service stability. Third, vehicles are tractable in terms of IP addresses, which can be mapped to location information and causes privacy concerns. Lastly, it is costly to offer Internet connectivity along all roads, especially in underdeveloped/remote areas.

On the other hand, we observe that CV applications often do not require a connection-based communication model. Those applications mainly need the information of geographic locations rather than vehicles. For example, the trip planning application on a vehicle just requires the traffic statuses (e.g., delays) along the specified route. Such a requirement can be efficiently satisfied by the named data networking (NDN), i.e., data distribution through the name. The data-driven, connection-less NDN architecture naturally supports in-network caching, built-in security, multi-path routing, and adaptive forwarding, which can effectively allow us to handle the scalability, mobility, and privacy concerns in supporting CV applications. Those facts motivate the direction of applying NDN in the CV application domain.

D. DATA MANAGEMENT MODEL

Before designing the CV application framework that can enable efficient data retrieval, we need to first identify how the data needed by CV applications are managed. We explain our thoughts regarding this issue in the following.

Since this paper focuses on CV applications built on data related to different geographical locations, we propose to distribute data to data servers (DSs) according to the location attribute. Specifically, we require that all data related to a basic geographic unit is always stored in one DS. The definition of a basic geographic unit is decided by the framework owner/designer. For example, suppose we are developing an NDN framework to support CV applications in a town with 8 basic geographic units (i.e., each of which can be a district in the town), denoted BGU_i, $i \in [0, 7]$. Then, we can establish 8 DSs and let each DS be responsible for the data of one basic geographic unit. Note that, the mapping of basic geographic units to DSs does not need to follow the 1-to-1 relationship. The data of multiple basic geographic units can be stored in one DS. This property allows us to easily balance the load over DSs under uneven data distribution.

Moreover, this model does not require all DSs to be centrally located in one place. Instead, they can be placed at different locations to enhance the scalability and efficiency of data distribution and storage. For example, a DS can be located in the same basic geographic unit whose data the DS is responsible for. Then, the data related to the geographic unit can be collected and stored in the DS timely without much transfer over the network, which saves the overload to the network. We also name each DS with its location and an index. For example, the t-th DS at location Loc_s is named "$/Loc_s/t$". Note that we follow the NDN naming convention to ensure the consistency with the data query (i.e., which will be shown in the next subsection). How to represent the location of a DS is introduced in Section IV-B.4.

In summary, we have the following definitions regarding data management in the proposed application framework.

- Data Allocation: all data related to a basic geographic unit is stored in the same DS.
- Data Server (DS) Naming: each DS is uniquely named by $/Loc_s/t$, where Loc_s represents the location it is physically located at and t is a unique sequence number within the location.

E. OVERALL CV APPLICATION FRAMEWORK DESIGN

With the above discussions, we propose an NDN based CV application framework as shown in Figure 1. The framework consists of the following core components.

- NDN Backbone: An NDN network that interconnects vehicles and data servers (DSs) for data retrieval. In practice, it can be established as an NDN overlay over the current network infrastructure.
- Data Servers (DSs): Servers that store data needed by CV applications. Note that, according to the discussion in Section III-D, data related to a basic geographic unit is always stored in the same DS. How geographic units are allocated to DSs for data storage are specified by the framework administrator.
- Location Name Service Server (LNSS): A server that maps a basic geographical unit to the DS that stores its data. The LNSS is designed to efficiently identify the DS to go to for data queries. The LNSS has a fixed location name for others to send requests to.
- Roadside Unit and Address Translator (RSU-ATs): Roadside devices that own two functions: 1) provide wireless connectivity to vehicles on road and connect them to the NDN backbone; and 2) attach the location of the DS that is responsible for storing the queried data as the forwarding hint [35] to Interest packets. RSU-ATs can be established by adding a mapping server over either existing wireless infrastructures such as LTE BSs/WiFi APs or dedicated DSRC units. Since each data retrieval usually is completed within a few seconds, we assume that each vehicle always receives the data, if fetched from the network successfully, through the same RSU-ATs that it sends the Interest packet to. This alleviates us from considering vehicle mobility when forwarding Interesting packets within the NDN backbone.

With such a framework, CV applications query the data of an area (i.e., target area) through the following procedures.

1) Generate the Interest packet that takes the name of the area as the name prefix followed by the information of the interest, i.e., $/[DataLocation]/[InterestInfo]$.
2) When the Interest packet arrives at the RSU-AT, the location of the DS that is responsible for storing the requested data (i.e., target DS or TDS) is attached to the packet as the forwarding hint, i.e., $/[DataLocation]/[InterestInfo](hint : [TDSLocation])$. The RSU-AT learns and maintains the mapping from *DataLocation* to *TDSLocation* from the LNSS. We purposely keep such a translation function to the roadside device for two reasons. First, this alleviates the load of vehicles. Second, the RSU-AT can more efficiently use the cached mapping information to serve vehicles passing by.
3) This packet is forwarded through the NDN backbone towards the target DS with the assistance of the forwarding hint. In this framework, NDN routers forward Interest packets first according to their forwarding hints.

In the above process, the location or identity of the vehicle is not included in the Interest packet, which keeps the privacy of vehicles. Figure 2 shows an example in which a vehicle wants to know the traffic status of location *Loc-A*. It then generates an Interest packet in the format of $/[Loc\text{-}A]/[traffic]$. When this packet arrives at the RSU-TA, it is translated into $/[Loc\text{-}A]/[traffic](hint : [Loc\text{-}A'])$, where $Loc\text{-}A'$ denotes the location of the target DS that stores the data related to *Loc-A*. The Interest packet is then forwarded inside the NDN backbone towards the target DS. The detailed design of the location name, i.e., *DataLocation* or *TDSLocation*, is explained later in Section IV-A.

The above application framework essentially turns the data query process into geographic routing. As shown later, this allows us to achieve efficient and scalable data query over the NDN architecture (i.e., see Section IV-C). Furthermore, though geographic routing is adopted, Interest packets still can take advantage of the caching capability of NDN to hit

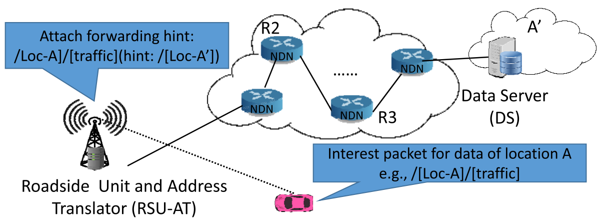

Attach forwarding hint: /Loc-A]/[traffic](hint: /[Loc-A'])
R2
NDN
......
NDN
A'
NDN
NDN
R3
Data Server (DS)
Roadside Unit and Address Translator (RSU-AT)
Interest packet for data of location A e.g., /[Loc-A]/[traffic]

FIGURE 2. A data query example in the trip planning application.

the requested data before reaching the target DS. Generally, the closer to the target DS, the more likely to find a cache of the requested data, as routers close to the target DS are more likely to have forwarded the requested data before.

It is not hard to see that, among the four major components, the DS, LNSS, and RAT-AT can be established by employing existing technologies. For example, the LNSS can be developed by requiring each DS to proactively report the areas that it covers and its location. The scalability of LNSS can be ensured through parallelism, i.e., by employing a cluster of servers, since the mapping of geographic locations to DSs is expected to be quite static. The RAT-AT can obtain the mapping information by querying the LNSS. Furthermore, since the DS and RAT-AT are distributed, they naturally own good scalability. Therefore, the **major challenge** in the application framework falls into how to **design the NDN backbone** to provide scalable and efficient data query service for CV applications. We present our efforts in solving this challenge in the next two sections.

IV. HIERARCHICAL HYPERBOLIC NDN BACKBONE ARCHITECTURE (H2NDN)

The NDN backbone interconnects CV applications and DSs. Considering the large number of vehicles and the vast areas they visit, the NDN backbone suffers a great pressure on scalability and efficiency. In this section, we present how we handle such a challenge when designing major components of the NDN backbone, namely the namespace, router topology, and Interest packet routing and FIB construction.

A. NAMESPACE

Since the proposed framework targets CV applications built upon information of different geographic locations, the name of each Interest or Data includes both the location information and data attributes. The general format contains two domains: $/[GeoLocation]/[Interest(Data)]$. The square bracket $[\cdots]$ indicates that it may contain multiple naming fields, as illustrated in the following.

The $/[GeoLocation]$ domain gives the name of the location related to the Interest or Data packet. We design this domain by following the hierarchical structure used in the postal address system. An example could be "$/country/state/city/road\_segment$". Such a design is motivated by two reasons. First, the hierarchical structure offers good scalability (i.e., like the postal system), which is necessary to support CV applications spanning a large area. Second, the name is self-explaining, which allows us to easily employ only static FIBs to forward Interest packets (Section IV-C). The $/[GeoLocation]$ domain is also used to name routers and DSs, which is introduced in Section IV-B.

The $[Interest(Data)]$ domain describes the attributes that precisely define the interested data. A general format includes three sub-domains: $/[Interest(Data)] := /[Time]/[Name]/[Filter]$. The $/[Time]$ sub-domain represents the time of interested data, e.g., "/10102018/morning". The $/[Name]$ sub-domain gives the specific name of the data such as "/congestion status" and "/weather". The $/[Filter]$ sub-domain describes the criteria to filter unwanted data. To support flexible application development, the last two domains are decided by CV application developers.

In practice, we assume that the administrator that oversees the NDN infrastructure is responsible for specifying the granularity of geographic location names and allocating/deciding names to applications (vehicles). The discussion of the detailed allocation scheme is out of the scope of this paper.

B. ROUTER TOPOLOGY AND NAMING

How routers are interconnected and organized affects the scalability and efficiency of processing data queries from CV applications. In this section, we first introduce two candidate topologies and analyze their limitations. We finally propose an H2 topology that handles the drawbacks, which is adopted in the application framework.

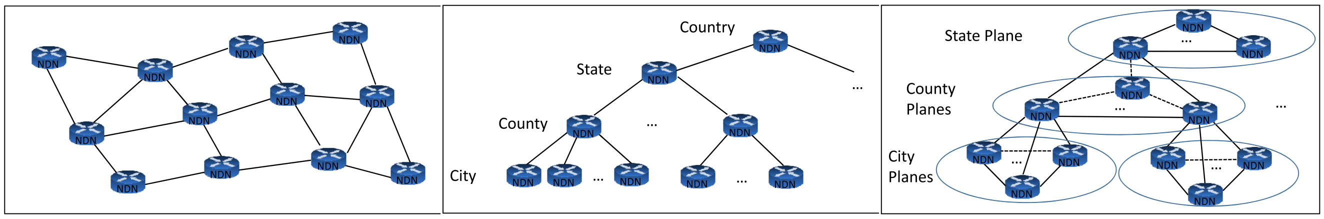
1) FLAT TOPOLOGY

The most intuitive idea is to deploy routers in a flat manner as shown in Figure 3a. In this topology, vehicles on road can connect to the NDN backbone through any NDN router (i.e., any router in Figure 3a). However, the flat topology has a limited ability to scale to a large area. This is because, under the flat layout, the hierarchical structure embedded inside geographic locations cannot be employed to guide the forwarding of Interest packets towards their target DSs. Thus, the number of FIB entries needed on each router grows in proportional to $O(N)$, where N is the total number of routers. Furthermore, it is also a formidable challenge to maintain the correctness of FIB entries when routers join and leave the framework dynamically.

Hyperbolic routing [31] can solve this issue by selecting the next hop based on hyperbolic coordinates. However, researches have shown that enabling hyperbolic routing at a large area (e.g., a whole state) is non-trivial and can easily suffer from sub-optimal routes and/or local minima [37].

2) HIERARCHICAL TOPOLOGY

The limitations of the flat topology drive the development of the hierarchical topology that integrates the hierarchical geographic location names into the router layout, as shown in Figure 3b. In this topology, each router covers a geographical area in a hierarchical manner. Thus, each router connects to all down-level routers under its area (denoted child routers)



(a) Flat topology. (b) Hierarchical topology. (c) Hierarchical Hyperbolic (H2) topology.

FIGURE 3. Illustration of possible NDN backbone topologies.

and the upper-level router that covers it (denoted parent router), if any. We further define edge routers as those on the lowest level in the topology, i.e., the city level in Figure 3b.

This topology makes it much easier to support data retrieval when Interest/Data follows the namespace defined in Section IV-A. Each NDN router just needs a few static FIB entries to forward Interest packets towards their target DSs. Specifically, upon receiving an Interest, the router can decide the forwarding option (i.e., "delivery to the associated DS", or "forward up one level to another router", or "forward down one level to another router") by comparing the name of the area covered by the router and the location name of the target DS. For example, when router covering $/countryA/stateB$ receives an Interest packet with targeting DS located at $/countryA/stateB/cityC/roadD/InterestE$, it can immediately learn that the packet should be forwarded down to the router covering $/countryA/stateB/cityC$.

However, the major drawback in the hierarchical topology is that high-level routers in this topology can easily get overloaded. For example, in Figure 3b, all cross-county Interests have to go through the state router that supervises these counties. Such a drawback limits the scalability and efficiency in supporting CV applications in large areas.

3) HIERARCHICAL HYPERBOLIC (H2) TOPOLOGY

In order to handle the drawbacks of the first two topologies, we combine them to design a novel hierarchical hyperbolic (H2) topology as shown in Figure 3c. This topology further groups same-level routers under the same parent router in the hierarchical topology as a hyperbolic plane. This results in many small hyperbolic planes. For example, as shown in Figure 3c, NDN routers covering cities under the same county form one hyperbolic plane. As a result, in addition to parent/child routers, each router further connects to neighbor routers in the same hyperbolic plane. As in the hierarchical topology, we also define NDN routers in the lowest layer in the H2 topology, e.g., those on city planes in Figure 3c, as edge routers. They connect to RSU-ATs that provide NDN connectivity to vehicles on road.

Hyperbolic routing is adopted on each hyperbolic plane to offload packets targeting destinations under the same parent router, which avoids the relaying through the parent router. This effectively reduces the load to high-level routers and thus improves the scalability of the hierarchical topology.

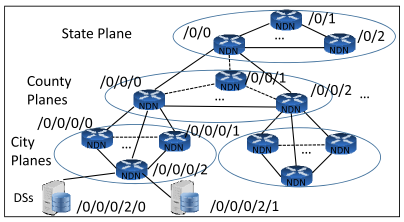


FIGURE 4. Router and DS name example.

Meanwhile, the limited size of each hyperbolic plane avoids the issue of low performance at a large scale (i.e., as explained when introducing the flat topology). We will show later that the routing of Interest/Data in the H2 topology can be achieved at the same efficiency as in the hierarchical topology (i.e., only requires a fixed amount of FIB entries).

4) ROUTER AND DS NAMING

In H2NDN, we also assign a name to NDN routers and DSs to facilitate the forwarding of Interest packets towards their target DSs. We only introduce how this is done in the H2 topology since it is the one adopted in the proposed framework. Specifically, we name every NDN router by the name of the geographic area it represents. Thus, except for the root router, every router inherits the name from the parent router in the H2 topology and adds a new field denoting the further split of the area. For example, the router that represents *city c* in *state b* of *country a* has a name of "$/country-a/state-b/city-c$".

Recall that we name each DS server by $/Loc_s/t$ in Section III-D, i.e., by its location and an index within the location. Since a router's name reflects its location, we thereby use the name of the immediately connected NDN router of a DS to denote its location. Figure 4 gives an example of the names for NDN routers and DS servers. Note that the two DSs connected to NDN router $/0/0/0/2$ are indexed by 0 and 1, respectively.

C. INTEREST PACKET ROUTING AND FIB CONSTRUCTION

With the above design, the proposed CV application framework searches requested data through geographic routing, i.e., routing Interest packets (i.e., which are received from edge routers) towards DSs that store the requested data (i.e., target DSs). While there are ongoing studies on designing routing protocols for discovering paths to data in NDN [35], [55], [56], we aim to attain efficient routing of Interest packets with only static FIB entries. This is critical for ensuring the scalability and efficiency of the H2NDN architecture. In the following, we first introduce the packet forwarding logic and the hyperbolic routing adopted by H2NDN. We then present the FIB construction method.

1) PACKET FORWARDING LOGIC

Recall that hyperbolic planes are added in the H2 router topology (i.e., Figure 3c). As a result, Interest packets may go through both hierarchical routing (i.e., up or down in the topology) and hyperbolic routing (i.e., over the hyperbolic plane) when traversing the NDN backbone to reach target DSs. This requires each router to dynamically decide the type of routing for each received Interest packet.

Our careful observations find that the above requirement can be satisfied by taking advantage of the hierarchical namespace. When an Interest packet arrives at a router, the routing type (i.e., hierarchical or hyperbolic) can be decided by comparing the geographic name of the router (i.e., denoted *HolderName*) with the geographic location of the target DS (i.e., denoted *TDSGeoName*). Note that the latter is stored inside the Interest packet as the forwarding hint (i.e., as introduced in Section III-E). Specifically, the longest matching prefix between *TDSGeoName* and *HolderName*, denoted by $LMPH$, can guide the forwarding action. We define the length of a name/prefix as the number of fields it contains. For example, the name "$/country-a/state-b$" has a length of 2. We use L_h to denote the length of the name of the current router, i.e. $L_h = HolderName.Length$. Then, the packet forwarding logic on each router is defined as:

- **Case 1:** If $LMPH.Length < L_h - 1$, this means that the target DS is out of the geographic area of the current router. Thus, the packet should be forwarded up one level to the parent of the current router.
- **Case 2:** If $LMPH.Length == L_h - 1$, this means that the target DS is in the geographic area of a router that is in the same hyperbolic plane with the current router. Thus, the packet should be forwarded in the hyperbolic plane through hyperbolic routing. We denote the current router as the entry router in the hyperbolic plane.
  The hyperbolic routing will forward the packet over the hyperbolic plane to the router that covers the area where the packet's target DS is located in. From that router, the packet will be forwarded through hierarchical routing towards the target DS. We thus define that router as the exit router of the interest packet in the hyperbolic plane.
- If $LMPH.Length == L_h$, this indicates that the target DS is under the geographic area of the current router. However, the current router may not be the one that directly connects to the target DS. We can distinguish this by comparing the length of the *TDSGeoName* and the $LMPH$. Particularly,
  - **Case 3:** If $TDSGeoName.Length == LMPH.Length$, this indicates that the target DS connects to the current router. Thus, the packet should be forwarded to the DS.
  - **Case 4:** If $TDSGeoName.Length > LMPH.Length$, this means that the target DS is not connected to the current router. Thus, the packet should be forwarded down to the child router that increases the length of $LMPH$.

When a DS receives an Interest packet, it checks whether the *TDSGeoName* matches with its name. If yes, it further checks whether a matched data can be found. Otherwise, the packet is dropped. If a matched data is found, it is sent back to the requester following standard NDN procedure.

2) HYPERBOLIC ROUTING

To implement the hyperbolic routing, we calculate the hyperbolic coordinates of routers on each hyperbolic plane by applying the HyperMap algorithm proposed in [33], [57] (i.e., note that HyperMap is also used in existing studies that evaluate the performance of hyperbolic routing in NDN [36]–[38]). Then, when an Interest packet enters the hyperbolic routing, we can greedily forward it towards its exit router in the hyperbolic plane over the hyperbolic space.

Note that the exit router of an Interest packet in a hyperbolic plane can be identified when it enters the plane by comparing the name of its target DS with the names of other routers in the hyperbolic plane. This is because the exit router should cover the area where the packet's target DS is located. Though this requires each router in the hyperbolic plane to store the names of all other routers, the small plane size in H2NDN makes such a cost acceptable. Moreover, the small plane size avoids problems faced by hyperbolic routing when the network size goes large [37].

3) FIB CONSTRUCTION METHOD

How to construct static FIB entries to implement the aforementioned packet forwarding logic is non-trivial. This is because, as shown above, the forwarding logic requires to compare the length of names, which cannot be directly supported by the current design of FIB in NDN. Fortunately, we find that this challenge can be indirectly solved by exploiting the rule of longest matching prefix in selecting the FIB entry to forward Interest packets. That is: when the *TDSGeoName* of an Interest packet matches with multiple prefixes, the entry of the longest prefix will be selected to forward the packet.

Particularly, we let FIB entries corresponding to the four cases in the forwarding logic own prefixes with different lengths. Thus, each packet will be processed only by the FIB entry corresponding to the case it belongs to. To assist the illustration, we use $HolderName[n]$ to represent the n-th field in *HolderName*. For example, if *HolderName* is $/A/B/C$, $HolderName[0]$ refers to the first field, i.e., A. Meanwhile, kindly note that the H2 topology has at least two layers. Thus,

TABLE 1. FIB construction template.

prefix	action
$/HolderName[0]$ (if $L_h \geq 3$)	to parent router
$/HolderName[0]/.../HolderName[L_h-2]$	hyperbolic routing
$/HolderName[0]/.../HolderName[L_h-1]$	to target DS
$ChildRouter1.Name$	to router 1
$ChildRouter2.Name$	to router 2
...	...
$ChildRouterX.Name$	to router X

TABLE 2. FIB entries of NDN router /0/0/0 in Figure 4.

prefix	action	example
/0	to parent router /0/0	*Interest*: /0/1/0/1
/0/0	hyperbolic routing	*Interest*: /0/0/2/1
/0/0/0/0	to child router /0/0/0/0	*Interest*: /0/0/0/0
/0/0/0/1	to child router /0/0/0/1	*Interest*: /0/0/0/1
/0/0/0/2	to child router /0/0/0/2	*Interest*: /0/0/0/2
...	...	...
$/0/0/0/x$	to child router $/0/0/0/x$	*Interest*: $/0/0/0/x$

the minimal L_h of all routers except the root is 2. We then design the FIB entries as the following.

Case 1 ($LMPH.Length < L_h - 1$): To capture packets that belong to this case, the corresponding FIB entry uses $/HolderName[0]$ as the matching prefix. The action is to forward the packet one level up to the parent router of the current router. Note that this entry does not exist when $L_h = 2$ (i.e., when the current router is on the highest level of the H2 topology), because all routers at this level belong to one hyperbolic plane, i.e., only hyperbolic routing is needed.

Case 2 ($LMPH.Length == L_h - 1$): To capture packets that belong to this case, the corresponding FIB entry uses the first $L_h - 1$ fields of the name of the current router (i.e., *HolderName*), i.e., $/HolderName[0]/\ldots/HolderName[L_h-2]$, as the matching prefix. The action is switching to hyperbolic routing (i.e., which is introduced in Section IV-C.2).

Case 3 ($LMPH.Length == L_h$ and $TDSGeoName.Length == LMPH.Length$:) To capture packets that belong to this case, the corresponding FIB entry takes the name of the current router (i.e., *HolderName*) as the matching prefix. The action is to deliver the Interest packet to the target DS connected to the current router.

Case 4 ($LMPH.Length == L_h$ and $TDSGeoName.Length > LMPH.Length$): For this case, we need a FIB entry for each of the child routers on the immediate lower level. The prefix of each entry is the name of the child router and the corresponding action is forwarding the Interest packet down to the child router. Note that in the adopted hierarchical namespace (i.e., Figure 4), the length of the name of a child router is one more than that of the current router (i.e., *HolderName*).

Table 1 summarizes the FIB entries described above. We show its validity through the following Lemma and proof.

Lemma 1: The proposed FIB prefixes can classify Interest packets into corresponding cases correctly and exclusively.

Proof: In Table 1, the first three rows match with Interests that fall into Cases 1, 2, and 3, respectively. All remaining rows in the table are for Case 4. The prefixes for the four cases have a length of 1, $L_h - 1$, L_h, and $L_h + 1$, respectively. We then have the following findings.

- Exclusiveness: Since the prefix(es) for each case has a different length, a packet will at most be classified into one case due to the rule of longest prefix match. This shows the exclusiveness of these FIB entries.
- Completeness: The prefix in the first row of the table, i.e., $/HolderName[0]$, is the name of the root node of the topology. According to the namespace design, it is the first field of all names. Thus, there must be at least one matching entry for any valid name. This shows the completeness of the designed entries.
- Correctness: The prefixes in the first three rows indicate $LMP.Lenth < L_h - 1$, $LMP.Lenth = L_h - 1$, and $LMP.Lenth = L_h = TDSGeoName.Length$, respectively, which are the definitions of the first three cases. Therefore, the three rows can classify packets that belong to the first three cases correctly. Further, a *TDSGeoName* that takes *ChildRouterX.Name* as the longest matching prefix makes $LMPH = HolderName$. Thus, it satisfies $LMPH.Length = L_h$ and $TDSGeoName.Length > LMPH.length$, which corresponds to Case 4. This shows the correctness. $\square$

The template shown in Table 1 can be easily implemented as long as each router knows the name of its parent and child routers (if any), which can be obtained through a protocol that exchanges names among neighbor routers. We can use it to develop FIB entries for any router in the H2NDN backbone. For example, the FIB of NDN router "/0/0/0" in Figure 4 can be implemented as Table 2. In this table, we also give out one example Interest name that falls into each entry.

V. CACHE ALLOCATION STRATEGY

The NDN architecture naturally supports routers to cache received Data for future Interests. Such a feature could further improve the efficiency of the proposed H2NDN backbone, as vehicles in the same or different locations may repetitively request the same data. However, by default, each NDN router adopts the First-In-First-Out (FIFO) strategy to replace caches when the cache space is full. Obviously, this un-coordinated strategy could easily create duplicate caches along the data forwarding path and thus cannot take full potential of available cache spaces. Therefore, we further study how to improve the cache efficiency of H2NDN.

In the following, we first model the problem and analyze its complexity and limits. We find that optimize data caching globally is impractical. We then propose a local algorithm that can greatly improve caching efficiency.

A. MODELING OF THE CACHE ALLOCATION PROBLEM

The goal of cache allocation is to maximize the saving of Interest/Data forwarding hops under the limited cache space on each router. For simplicity, we use Interest/Data to denote

Interest packet/Data packet in this section. Then, the problem can be modeled as the following:

$$\text{maximize} \sum_{r\in\mathbb{R}}\sum_{i\in\mathbb{I}} f_{ri}b_{ri}\mathbb{H}(r,\mathbb{P}(i)) \tag{1}$$

$$\text{subject to } \forall r\in\mathbb{R} \quad \sum_{i\in\mathbb{I}} b_{ri}x_i \le s_r \tag{2}$$

$$f_{ri} = \begin{cases} \sum\limits_{n\in\mathbb{N}(r,\mathbb{P}(i))} f_{ni}\overline{b}_{ni}, & \text{if } r \notin \mathbb{E} \\ Input(r,i) & \text{if } r\in\mathbb{E} \end{cases} \tag{3}$$

$$b_{ri}\in\{0,1\}, \quad \overline{b}_{ri} = 1 - b_{ri} \tag{4}$$

where $\mathbb{I}$ and $\mathbb{R}$ denote the set of all Interests and routers, respectively, f_{ri} represents Interest I_i's visiting frequency at router R_r, b_{ri} is a binary value to denote whether R_r owns a copy of the data that matches I_i, x_i is the size of I_r, s_r denotes the cache space of router R_r, $\mathbb{E}$ is the set of edge routers. Further, function $\mathbb{P}(i)$ returns the DS that stores the data that matches I_i (i.e., I_i's target DS), $\mathbb{H}(r,\mathbb{P}(i))$ returns the number of hops from router R_r to DS server $\mathbb{P}(i)$, function $\mathbb{N}(r,\mathbb{P}(i))$ returns the set of routers that 1) are direct neighbors of R_r and 2) take R_r as the next hop on the path to $\mathbb{P}(i)$, and function $Input(r,i)$ returns edge router R_r's frequency of receiving Interest I_i from vehicles passing by.

In the above modeling, we exploit the fact that Interests enter the NDN backbone through edge routers (i.e., we use $\mathbb{E}$ to denote the set of all edge routers), from where they are forwarded towards their target DSs through the H2NDN. Meanwhile, Interest packets are stopped from further forwarding whenever they hit the requested Data. Therefore, we model Interest I_i's visiting frequency at a non-edge router R_r as Equation (3), i.e., the sum of the visiting frequencies at neighbor routers that do not have the matching Data and take R_r as the next hop to the target DS of I_i.

B. IMPRACTICALITY OF GLOBAL OPTIMIZATION

Even with the modeling of the cache allocation problem, it is impractical to conduct global cache allocation optimization the H2NDN backbone due to two reasons.

First, the problem is NP-hard. We can show that a simple case of the problem (i.e., with only two NDN routers) is a 0-1 linear programming problem, which is NP-hard. The detailed proof of the NP-hardness is presented in Appendix A. Second and more importantly, the global cache allocation needs to establish a central server that can collect the Interest visiting frequencies from all routers timely. It also needs to notify the cache plans to all routers. Clearly, such a design does not scale well with the size of the H2NDN backbone. Note that the proposed application framework is expected to cover a large area with a large number of routers. Thus, the centralized optimization approach can easily become a performance bottleneck and a single point of failure.

C. LOCAL CACHING METHOD

Consequently, we further propose a local caching method that can also greatly improve caching efficiency.

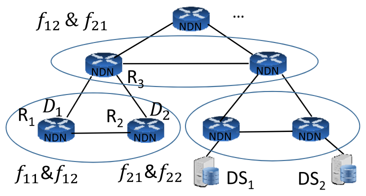


FIGURE 5. Effect of the visiting frequency aggregation (i.e. R_3 aggregates the visiting frequencies of Interests that are not satisfied by R_1 and R_2).

1) HIGH-LEVEL IDEA

Our idea exploits the modeling of the return of caching a piece of Data and the Interest visiting frequency at each router (i.e., Eqs (1) and (3)).

Particularly, first, if a router cannot satisfy a particular Interest, the Interest's visiting frequency to this router is aggregated to the next-hop router on the path to its target DS. We show this phenomenon in a simplified H2 topology in Figure 5, in which router R_1 and R_2 only caches D_1 (i.e., which is stored at DS_1) and D_2 (i.e., which is stored at DS_2), respectively. In the example, we also assume that R_1 (or R_2) receives Interests requesting for D_1 and D_2 at a frequency of f_{11} and f_{12} (or f_{21} and f_{22}), respectively. We see that R_3 only aggregates the visiting frequencies of Interests that are not satisfied by R_1 and R_2 (i.e., f_{12} and f_{21}). Thus, we have the following observation:

O1 The closer a router is with the target DS of an Interest, the more chances for the router to aggregate visiting frequencies from the Interest.

Second, the benefit of caching the Data for Interest I_i at router R_r can be calculated by multiplying the visiting frequency with the number of saved hops, i.e., $f_{ri} * \mathbb{H}(r,\mathbb{P}(i))$. This indicates the following:

O2 The closer the Data cache is placed to the target DS of an Interest, the fewer forwarding hops it can save under the same amount of visiting frequency.

By combining observations *O1* and *O2*, we see that it is desirable to cache popular Data away from DSs and less-popular data close to DSs. In this way, popular Data maximizes the number of saved forwarding hops, while less-popular Data aggregates more visiting frequencies. To achieve this effect, we propose the following strategy:

- All Data arriving at a router competes for cache spaces based on their expected hop savings.
- The expected hop saving of a Data cache is calculated based on the Interest visiting frequency measured at the router and the distance to the target DS.

In this process, popular Data initially wins the competition on all routers due to their high visiting frequencies. However, their caches on edge routers (or those close to edge routers) capture Interests and thus lower their visiting frequencies at routers close to DSs. This makes less-popular Data able to win at a place where their aggregated visiting frequency surpasses that of popular Data. Consequently, popular Data sinks towards edge routers and less-popular Data pops up towards DSs, which is exactly what we hope to attain.

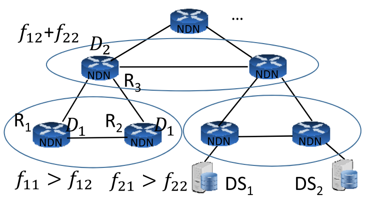


FIGURE 6. Routers R_1, R_2, and R_3 cache D_1, D_1, and D_2, respectively, after the local Data competition (assume each router has only one cache slot).

Figure 6 shows the effect of such a local competition in the same configuration as Figure 5. We again assume that each router can cache at most one piece of Data. We suppose that $f_{11} > f_{12}$ and $f_{21} > f_{22}$. Then, initially, D_1 wins D_2 on all three routers (i.e., R_1, R_2, and R_3) and is cached at those routers. However, as long as D_1 is cached at R_1 and R_2, the cache captures subsequent Interests for D_1. This makes R_3 only receive Interests for D_2. As a result, D_2 wins the competition at R_3 and is cached there. We can see that the above process essentially sinks D_1 towards edge routers and pops D_2 towards the target DS. It is also not hard to see that the resulted caching plan in this example reaches optimal.

We discuss the details to implement the above high-level idea in the following subsections.

2) INTEREST VISITING FREQUENCY COLLECTION

Each router updates the visiting frequencies of Interests periodically. Particularly, we first define a period T. Then, each router updates the visiting frequency of Interest I_i, denoted by VF_i, at the end of each period by the following:

$$VF_i^k = \alpha * MF_i^k + (1 - \alpha) * VF_i^{k-1} \tag{5}$$

where k is the index of the period, $MF_i^{T_k}$ denotes the visiting frequency of Interest I_i measured in the k-th period, and $\alpha \in (0, 1]$ is a weighting factor. Obviously, α reflects the aggressiveness in tracking the change of visiting frequencies. The larger α is, the more quickly it responds to the change of Interest visiting frequency. However, a large α also leads to frequent fluctuation of measured visiting frequencies. We thereby set it to 0.5 by default in this paper.

3) DATA COMPETITION PROCESS

The data competition process is conducted dynamically. Specifically, whenever a piece of new Data arrives at a router, the router first checks whether the cache space is full or not. If not, the Data is cached directly. Otherwise, the router calculates the Data's expected number of hop savings and compare it with that of Data already cached on the router. Then, the Data cache with the minimal hop saving is removed from the router. Meanwhile, at the end of each visiting frequency collection period, the hop saving of each cached Data is updated based on the updated visiting frequency. Table 3 gives an example of the information on each router.

4) OVERALL CACHE ALLOCATION ALGORITHM

We summarize the cache allocation algorithm running over each router as Algorithm 1. We see that the complexity of the cache allocation subroutine (i.e., the *AllocateCacheFor*(*Data d*) function in Algorithm 1) is $O(M)$, where M is the size of the cache table. Meanwhile, the complexity for interest visiting frequency update is $O(N_i)$, where N_i is the number of received interest packets. Since both procedures own linear complexity, we think the proposed algorithm can be practically deployed.

Algorithm 1 Cache Allocation Algorithm on Each Router

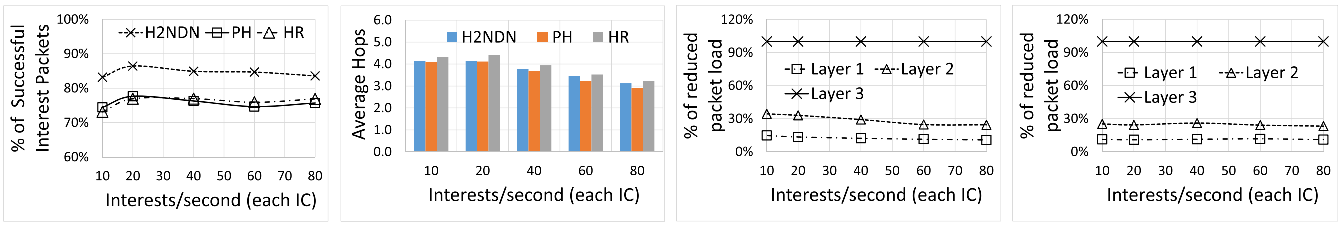
```
Function RecordInterest (Interest $I_i$):
    $MF_i = MF_i + 1$;
End Function
Function UpdateInterestVisitFreq (int $k$):
    foreach $I_i \in \mathbb{I}$ do
        $VF_i = \alpha * MF_i + (1 - \alpha) * VF_i$;
        $MF_i = 0$;
    end
End Function
Function AllocateCacheFor (Data $d$):
    if $CacheTable.size() < CacheTable.Limit()$ then
        $CacheTable.Insert(d)$;
    end
    else
        $d_{min} = CacheTable.ReturnMin()$;
        if $d.HopSaving > d_{min}.HopSaving$ then
            $CacheTable.Erase(d_{min})$;
            $CacheTable.Insert(d)$;
        end
    end
End Function
Function Main ():
    k=0;
    Every $T$ amount of time
      $UpdateInterestVisitFreq(k)$;
      k=k+1;
    For each received Interest $I$
      $RecordInterest(I)$;
    For each received Data $d$
      $AllocateCacheFor(d)$;
End Function
```

TABLE 3. Example information table on a router for data competition.

Interest (Data)	Visiting Freq.	Hop to Target DS	Hop Saving
I_1	200	10	2000
I_20	160	8	1280
I_5	120	10	1200
...	...	...	...
I_100	30	5	150

VI. PERFORMANCE EVALUATION

Since this paper primarily develops the H2NDN backbone and the cache allocation strategy, we mainly evaluate the two components in this section. When evaluating



(a) Interest packet success rate. (b) Average number of hops. (c) H2NDN's Load reduction over PH on layer-1, -2, -3 routers. (d) H2NDN's Load reduction over HR on layer-1, -2, -3 routers.

FIGURE 7. Performance of H2NDN in the experiments with the Roma trace.

the performance of H2NDN, we adopted two comparison approaches: the hierarchical routing over the pure hierarchical topology (i.e., denoted PH) [30] and the greedy hyperbolic routing over the hierarchical hyperbolic (H2) topology (i.e., denoted HR) [36]. In this test, the cache space on each router is set to 20 (i.e., can cache 20 pieces of Data), and the least-recently-used (LRU) cache allocation policy is adopted. After showing the advantages of H2NDN, we compare the performance of the proposed cache allocation strategy with the LRU strategy and the ProbCache algorithm [42].

We adopted the NS3-based ndnSIM [58] as the experiment tool. In the experiment, we assume the trip planning application in which vehicles periodically query the statuses of interested places such as traffic and weather. We adopted a real trace (i.e., Roma taxi trace [59]) and a random trace to drive the generation of queries. The Roma trace provides GPS records of 320 taxis in the Roma city for one month in 2014. Thus, in the experiment with the Roma trace, vehicles query for places that they are going to visit in the next hour (which is obtained by pre-processing the trace). In the test with the Random trace, vehicles simply query for information of randomly selected places.

We build a 3-layer H2 topology over the area covered in the Roma trace. We set the number of layers to 3 because the generated topology is enough to cover the area visited by vehicles in the Roma trace (i.e., not making a high-level router have too many child routers). On the highest layer (i.e., layer 2), the whole map is divided into 9 sub-areas evenly (i.e., 3×3), and each sub-area is represented by one NDN router. The area of each layer-2 NDN router is split into 9 sub-areas on layer 1 (i.e., 3×3), each of which is also represented by one NDN router. The area of each layer-1 router is split into 2km by 2km sub-areas on layer 0, i.e., the number of sub-areas varies depending on the area size. On layer 2, all routers form a hyperbolic plane. On layer 1 or 0, NDN routers with the same parent NDN router form a hyperbolic plane. Meanwhile, to evaluate the PH architecture, we adapted the H2 topology to generate the PH topology. Particularly, we added a layer 3 NDN router that covers all layer 2 NDN routers and ignored all hyperbolic planes (i.e., as shown in Figures 3b and 3c). Every link in both topologies has 1 Mbps bandwidth.

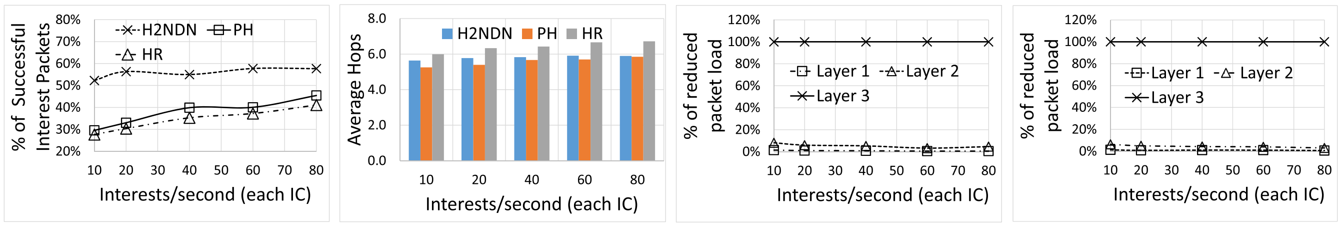
In both H2 topology and PH topology, every edge NDN router (i.e., layer-0 router) is attached with one DS server and one interest consumer (IC). We assume that each DS stores data related to the area covered by the immediate edge router it connects. Such data is to be queried by vehicles. Each IC collects Interest packets from vehicles in the area of the corresponding layer-0 NDN router and sends them to the H2NDN backbone to retrieve interested data.

In the experiment, we measured the success rate and the average number of hops (i.e., delay) of Interest packets. The former refers to the percentage of Interest packets that successfully retrieve the queried data. The latter represents the average number of hops needed by Interest packets to hit the queried data. We also measure the load of NDN routers as the number of Interest packets they receive. The three metrics reflect the performance of the H2NDN and the cache allocation strategy in supporting CV applications.

A. PERFORMANCE EVALUATION WITH THE ROMA TRACE

In this test, we varied the interest generation rate on each IC from 10/s to 80/s and measured the data retrieval performance. We chose such a range since each IC receives Interest packets from all vehicles passing by, and each vehicle may issue a few interest queries when connected with an IC. The target DSs of Interest packets are generated by following the mobility of taxis in the Roma trace. The results are shown in Figures 7a, 7b, 7c, and 7d. We see from Figure 7a that PH and HR achieve roughly the same interest query success rate, while H2NDN presents a much higher success rate. This shows that hyperbolic routing (HR) can efficiently approximate the performance of the hierarchical routing (PH) that follows the shortest path to reach target DSs (i.e., as illustrated in Section IV-B.2). However, they both tend to overload high-level routers, which leads to a lot of packet losses. H2NDN effectively solves this issue by offloading the load of high-level routers through purposely-designed hyperbolic planes, which improves the success rate.

We find from Figure 7b that the average number of hops experienced by successful packets is very close in H2NDN and PH, while that in HR is obviously higher. Technically, the PH always uses the shortest path to forward an interest packet to the target DS, while the design of the hyperbolic plane in H2NDN increases the number of hops needed. However, the routing over hyperbolic planes improves the utilization of router caches, as each router also forwards and



(a) Interest packet success rate. (b) Average number of hops. (c) H2NDN's Load reduction over PH on layer-1, -2, -3 routers. (d) H2NDN's Load reduction over HR on layer-1, -2, -3 routers.

FIGURE 8. **Performance of H2NDN in the experiments with the random trace.**

caches Data from routers on the same plane (rather than only from parent or child routers). Therefore, the average number of hops of successful Interest packets in H2NDN is similar to that in PH. On the other hand, as shown in past studies, hyperbolic routing cannot always find the shortest path to the destination and thus owns the largest average number of hops.

We further see from Figures 7c and 7d that H2NDN effectively reduces the load of high-level routers. The percentages of reduction to layer-1, layer-2, and layer-3 routers are 12%, 30%, and 100% when compared with PH and 10%, 25%, and 100% when compared with HR. Since most Interest packets arriving at layer-0 routers are not confined to nearby places, most of them are forwarded to layer-1 routers that cover a larger area in both PH and H2 topologies. Thus, H2NDN does not save much load on layer-1 routers when compared with PH and HR. However, quite many Interest packets target DSs that are located within the same layer-2 router (i.e., since Interest packets are generated for places that vehicles are going to visit in the next hour). Those packets are handled by hyperbolic routing on layer-1 in H2NDN without being forwarded to layer-2 routers, which effectively reduces the load to layer-2 routers. Further, H2NDN does not need layer-3 routers, which makes it save 100% of the load to layer-3 routers. We also see that H2NDN's saving over HR is smaller than that over PH. This is because HR can also employ same-layer links to forward interest packets (which is not adopted in PH). Such a result s consistent with the result on interest query success rate and shows that H2NDN can effectively offload the load to high-level routers.

B. PERFORMANCE EVALUATION WITH THE RANDOM TRACE

In this test, Interest packets are generated randomly at each IC. The results are shown in Figures 8a, 8b, 8c, and 8d. We find that the results are consistent with those in the previous test with trave-driven packet generation. The H2NDN leads to a better performance than PH and HR, while the performances of PH and HR are similar.

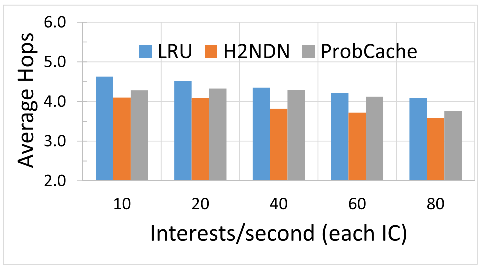
First, we notice that the Interest packet success rates of all the three approaches (i.e., H2NDN, PH, and HR) are lower in this test than those in the test with the Roma trace. This is because all ICs generate Interest packets in the Random trace, while some ICs in the Roma trace do not generate packets if no taxis pass by during the testing period. Thus, many more Interest packets are generated in this test, which causes a higher packet loss rate in the tests with all approaches. However, we still see that H2NDN performs much better than PH and HR. Second, we see that H2NDN saves less on layer-1 and layer-2 routers than in the test with the Roma trace. This is because the random Interest packet generation in this test creates more Interest packets destined to far-away DSs than in the test with the Roma trace. Those packets have to be forwarded to layer-2 routers even in H2NDN. However, H2NDN still saves some load for layer-1 and layer-2 routers and all the load for the layer-3 router.

By summarizing the above experiment results, we conclude that the proposed H2NDN architecture is efficient in offloading load to high-level routers and thus improves the scalability and routing performance. This is meaningful in supporting data retrieval for CV applications.

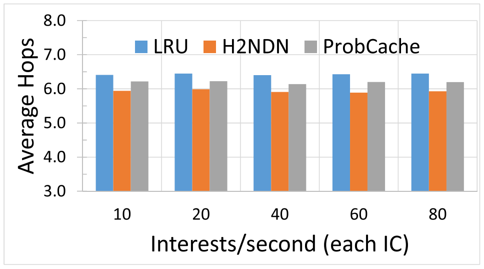
C. EVALUATION OF THE CACHE ALLOCATION STRATEGY

We further evaluated the performance of the distributed cache allocation strategy proposed in Section V. We compared it with the least-recently-used (LRU) policy used by ndnSIM and the ProbCache algorithm proposed in [42]. We also use H2NDN to represent our caching strategy in this subsection. Note that other cache policies in ndnSIM (such as priority-FIFO) present similar results as LRU and thus are not presented. We used the average number of hops as the evaluation metric. For fairness, we adopted the H2 backbone topology in evaluating the three approaches. We set the cache space on each router to 10 and 20 in this experiment. We chose the two numbers because they lead high and moderate cache replacement frequencies (i.e., severe and moderate cache space shortage) in the experiments, respectively, which are needed to demonstrate the difference between approaches adopted in this experiment. The experiment results are plotted in Figure 9 and Figure 10.

We see from Figures 9a and 9b that H2NDN presents the lowest number of hops needed by Interest packets to hit the queried data under both the Roma trace and the Random trace. The major reason is that H2NDN considers both the saving of forwarding hops and the popularity of data when creating caches. On the other hand, LRU only considers the local data popularity and fails to consider the saving of forwarding hops, while ProbCache fails to consider data popularity (i.e., it considers the cache space availability and

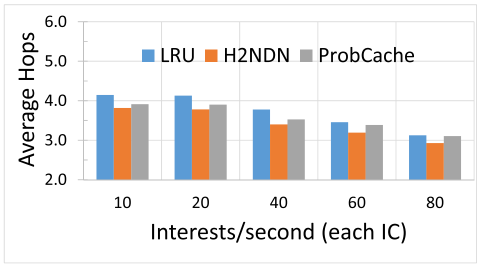


(a) Roma trace.

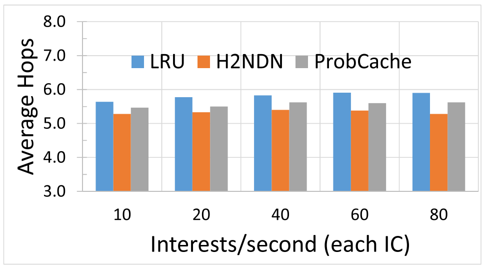


(b) Random trace.

FIGURE 9. Evaluation of the cache allocation strategy (cache space is 10).



(a) Roma trace.



(b) Random trace.

FIGURE 10. Evaluation of the cache allocation strategy (cache space is 20).

the percentage of hop saving). As a result, both LRU and ProbCache cannot optimize the cache replacement towards maximizing the number of saved hops. We also see that ProbCache presents fewer average number of hops than LRU. This is because, by treating routers with different distance to the data source differently, it avoids repetitively creating the same cache along the forwarding path.

We further notice that H2NDN saves fewer hops over LRU and ProbCache in the experiment with the Random trace than with the Roma trace. This is because the generation of Interest packets presents a relatively stable pattern in the Roma trace, which allows H2NDN to reliably create caches for data that can save more forwarding hops.

The results in Figures 10a and 10b (i.e., cache space 20) are consistent with those in Figures 9a and 9b (i.e., cache space 10). We further notice that the percentage of saving is lower when the cache space is set to 20. This is because, the more cache space each NDN router owns, the less likely that the LRU strategy and the ProbCache algorithm fail to cache the data that has a high potential in saving forwarding hops. However, we notice that H2NDN still presents around 10% improvement over LRU and 5% over ProbCache, which further validates its effectiveness in caching data.

Combining the above results, we conclude that the proposed cache allocation strategy in H2NDN can effectively utilize cache spaces on NDN routers. Meanwhile, as explained in Section V, it owns a low complexity and works in a distributed manner. Such characteristics enable H2NDN to effectively support CV applications.

VII. CONCLUSION

In this paper, we propose an NDN based framework to support CV applications after recognizing the potential synergy between NDN and CV applications. Considering the large number of vehicles and the vast areas they visit, we propose to enhance the efficiency and scalability of the framework in two aspects. We first propose a novel hierarchical hyperbolic NDN backbone architecture named H2NDN. It takes advantage of the hierarchical structure of geographic location names to build a hierarchical namespace and router topology. It carefully defines hyperbolic planes in the hierarchical architecture to offload the traffic to high-level routers. As a result, only static FIB entries are needed to forward Interest packets in the architecture. Those features make the H2NDN backbone present high efficiency and scalability in supporting CV applications. We further propose an advanced distributed cache allocation strategy that can improve the efficiency of cache spaces on NDN routers. Extensive real trace-based NS3 simulation demonstrates the efficiency of the proposed architecture and algorithm. In the future, we plan to exploit the mobility patterns of vehicles to enhance the performance of Interest packet routing and Data caching in the NDN backbone.

A. PROOF OF NP-HARDNESS OF THE CACHE ALLOCATION PROBLEM

To show the NP-hardness, we first consider a special case of the cache allocation problem modeled in Eqs. (1)-(4). As shown in Figure 11, the special case consists of two NDN routers that are directly connected. Meanwhile, each NDN router is attached with one DS and receives Interest packets targeting both DSs from vehicles passing by. We assume that router R_1 receives Interests for DS_1 and DS_2 at the frequency of f_{1i}^1 and f_{2j}^1, respectively, where i and j are indexes of Data stored at the two DSs, respectively. Similarly, R_2 receives Interests for the two DSs at the frequency of f_{1i}^2 and f_{2j}^2,

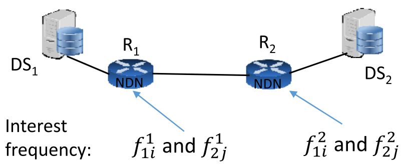


FIGURE 11. A simplified case of the cache allocation problem.

restrictively. In this case, the cache allocation problem can be rewritten as the following.

$$\text{maximize} \quad \begin{aligned}&(f_{1i}^1 + f_{1i}^2(1-b_{2i})) * b_{1i} * 1 + f_{1i}^2 * b_{2i} * 2\\ &+(f_{2j}^2 + f_{2j}^1(1-b_{1j})) * b_{2j} * 1 + f_{2j}^1 * b_{1j} * 2\end{aligned} \tag{6}$$

$$\text{subject to} \sum_i b_{1i}x_i + \sum_j b_{1j}x_j \le s_1 \tag{7}$$

$$\sum_i b_{2i}x_i + \sum_j b_{2j}x_j \le s_2 \tag{8}$$

$$\forall r \in \{1,2\}, i,j \quad b_{ri} \in \{0,1\},\ b_{rj} \in \{0,1\} \tag{9}$$

where b_{ri} (or b_{rj}) is a binary number that denotes whether router R_r has a cache of Data indexed by i (or j). In Equation (6), $(f_{1i}^1 + f_{1i}^2(1-b_{2i})) * b_{1i} * 1 + f_{1i}^2 b_{2i} * 2$ and $(f_{2j}^2 + f_{2j}^1(1-b_{1j})) * b_{2j} * 1 + f_{2j}^1 b_{1j} * 2$ denote the number of hop savings at router R_1 and R_2, respectively. The equation can be re-organized as the following:

$$\begin{aligned}&(f_{1i}^1 + f_{1i}^2) * b_{1i} * 1 + 2 * f_{1i}^2 * b_{2i} - f_{1i}^2 * b_{1i} * b_{2i}\\ &\quad + (f_{2j}^2 + f_{2j}^1) * b_{2j} * 1 + 2 * f_{2j}^1 * b_{1j} - f_{2j}^1 * b_{1j} * b_{2j}\end{aligned} \tag{10}$$

Equation (10) is non-linear due to the multiplication: $b_{1i} * b_{2i}$ and $b_{1j} * b_{2j}$. Fortunately, we can remove the non-linearity by replacing $b_{1i} * b_{2i}$ and $b_{1j} * b_{2j}$ with binary variables z_i and z_j, respectively, when the following limits are satisfied.

$$\begin{aligned}&z_i \le b_{1i}, \quad z_i \le b_{2i}\\ &z_i \ge b_{1i} + b_{2i} - 1\\ &z_j \le b_{1j}, \quad z_j \le b_{2j}\\ &z_j \ge b_{1j} + b_{2j} - 1\end{aligned}$$

We can easily verify that z_i and z_j are equivalent to $b_{1i} * b_{2i}$ and $b_{1j} * b_{2j}$, respectively, with the above constrains.

Consequently, Eqs. (6)-(9) turn to the following:

$$\begin{aligned}\text{maximize} \quad &(f_{1i}^1 + f_{1i}^2) * b_{1i} * 1 + 2 * f_{1i}^2 * b_{2i} - f_{1i}^2 * z_i\\ &+(f_{2j}^2 + f_{2j}^1) * b_{2j} * 1 + 2 * f_{2j}^1 * b_{1j} - f_{2j}^1 * z_j\\ \text{subject to} \quad &\sum_i b_{1i}x_i + \sum_j b_{1j}x_j \le s_1\\ &\sum_i b_{2i}x_i + \sum_j b_{2j}x_j \le s_2\\ &\forall r \in \{1,2\}, \quad i,j \quad b_{ri} \in \{0,1\},\ b_{rj} \in \{0,1\}\\ &\forall i,j \quad z_i \in \{0,1\}, \quad z_j \in \{0,1\}\\ &z_i \le b_{1i}, \quad z_i \le b_{2i}\\ &z_i \ge b_{1i} + b_{2i} - 1\\ &z_j \le b_{1j}, \quad z_j \le b_{2j}\\ &z_j \ge b_{1j} + b_{2j} - 1\end{aligned}$$

The above modeling is a classical 0-1 linear programming problem and thus is NP-hard. Such a fact proves that the global cache allocation problem modeled in Eqs. (1)-(4) is NP-hard. Otherwise, if the global cache allocation problem owns a polynomial-time solution, the special case can be solved in polynomial-time. This contradicts with the fact that the special case is proved to be NP-hard.

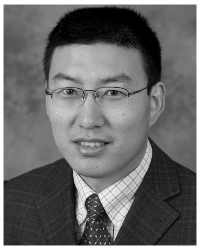
REFERENCES

[1] M. Faezipour, M. Nourani, A. Saeed, and S. Addepalli, "Progress and challenges in intelligent vehicle area networks," *Commun. ACM*, vol. 55, no. 2, pp. 90–100, Feb. 2012.
[2] *Connected Vehicle Basics*. Accessed: 2018. [Online]. Available: https://www.its.dot.gov/cv_basics/cv_basics_what.htm
[3] *Connected Vehicle Applications*. Accessed: 2018. [Online]. Available: https://www.its.dot.gov/pilots/cv_pilot_apps.htm
[4] *Connected Vehicle Pilot Deployment Program*. Accessed: 2018. [Online]. Available: https://www.its.dot.gov/pilots/
[5] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, T. Abdelzaher, L. Wang, P. Crowley, E. Yeh, and C. Papadopoulos, "Named data networking (NDN) project," Xerox Palo Alto Res. Center-PARC, Palo Alto, CA, USA, PARC Tech. Rep. 2010-003, 2010, p. 158, vol. 157.
[6] M. Amadeo, C. Campolo, and A. Molinaro, "Information-centric networking for connected vehicles: A survey and future perspectives," *IEEE Commun. Mag.*, vol. 54, no. 2, pp. 98–104, Feb. 2016.
[7] D. Grewe, M. Wagner, M. Arumaithurai, I. Psaras, and D. Kutscher, "Information-centric mobile edge computing for connected vehicle environments: Challenges and research directions," in *Proc. Workshop Mobile Edge Commun. (MECOMM)*, 2017, pp. 7–12.
[8] M. Amadeo, C. Campolo, and A. Molinaro, "CRoWN: Content-centric networking in vehicular ad hoc networks," *IEEE Commun. Lett.*, vol. 16, no. 9, pp. 1380–1383, Sep. 2012.
[9] M. Chen, D. Ong Mau, Y. Zhang, T. Taleb, and V. C. M. Leung, "VENDNET: VEhicular named data NETwork," *Veh. Commun.*, vol. 1, no. 4, pp. 208–213, Oct. 2014.
[10] M. Kuai, X. Hong, and R. R. Flores, "Evaluating interest broadcast in vehicular named data networking," in *Proc. 3rd GENI Res. Educ. Exp. Workshop*, Mar. 2014, pp. 77–78.
[11] Y.-T. Yu, Y. Li, X. Ma, W. Shang, M. Y. Sanadidi, and M. Gerla, "Scalable opportunistic VANET content routing with encounter information," in *Proc. 21st IEEE Int. Conf. Netw. Protocols (ICNP)*, Oct. 2013, pp. 1–6.
[12] M. Amadeo, C. Campolo, and A. Molinaro, "Enhancing content-centric networking for vehicular environments," *Comput. Netw.*, vol. 57, no. 16, pp. 3222–3234, Nov. 2013.
[13] S. H. Ahmed, S. H. Bouk, M. A. Yaqub, D. Kim, H. Song, and J. Lloret, "CODIE: Controlled data and interest evaluation in vehicular named data networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3954–3963, Jun. 2016.
[14] S. H. Ahmed, S. H. Bouk, M. A. Yaqub, D. Kim, and H. Song, "DIFS: Distributed interest forwarder selection in vehicular named data networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 9, pp. 3076–3080, Sep. 2018.
[15] A. Boukerche, R. W. L. Coutinho, and X. Yu, "LISIC: A link stability-based protocol for vehicular information-centric networks," in *Proc. IEEE 14th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Oct. 2017, pp. 233–240.
[16] G. Grassi, D. Pesavento, G. Pau, L. Zhang, and S. Fdida, "Navigo: Interest forwarding by geolocations in vehicular named data networking," in *Proc. IEEE 16th Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2015, pp. 1–10.
[17] X. Li, S. Wang, W. Wu, X. Chen, and B. Xiao, "Interest tree based information dissemination via vehicular named data networking," in *Proc. 27th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2018, pp. 1–9.

[18] Z. Lin, M. Kuai, and X. Hong, "Reliable forwarding strategy in vehicular networks using NDN," in *Proc. IEEE 84th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2016, pp. 1–5.
[19] J. M. Duarte, T. Braun, and L. A. Villas, "MobiVNDN: A distributed framework to support mobility in vehicular named-data networking," *Ad Hoc Netw.*, vol. 82, pp. 77–90, Jan. 2019.
[20] G. Deng, L. Shi, R. Li, and X. Xie, "Efficient inter-vehicle Internet content distribution based on named data," in *Proc. IEEE 82nd Veh. Technol. Conf. (VTC-Fall)*, Sep. 2015, pp. 1–5.
[21] T. Jiang, X. Xu, L. Pu, Y. Hu, and Z. Qiu, "A simulation study of connected vehicle systems using named data networking," in *Proc. Int. Conf. Cloud Comput.* Cham, Switzerland: Springer, 2013, pp. 39–48.
[22] J. Wang, R. Wakikawa, and L. Zhang, "DMND: Collecting data from mobiles using named data," in *Proc. IEEE Veh. Netw. Conf.*, Dec. 2010, pp. 49–56.
[23] D. Grewe, S. Schildt, M. Wagner, and H. Frey, "ADePt: Adaptive distributed content prefetching for information-centric connected vehicles," in *Proc. IEEE 87th Veh. Technol. Conf. (VTC Spring)*, Jun. 2018, pp. 1–5.
[24] D. Grewe, M. Wagner, and H. Frey, "PeRCeIVE: Proactive caching in ICN-based VANETs," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Dec. 2016, pp. 1–8.
[25] N. Abani, T. Braun, and M. Gerla, "Proactive caching with mobility prediction under uncertainty in information-centric networks," in *Proc. 4th ACM Conf. Inf.-Centric Netw. (ICN)*, 2017, pp. 88–97.
[26] G. Mauri, M. Gerla, F. Bruno, M. Cesana, and G. Verticale, "Optimal content prefetching in NDN vehicle-to-infrastructure scenario," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2513–2525, Mar. 2017.
[27] Y. Kurihara, Y. Koizumi, and T. Hasegawa, "Compact data structures for location-based forwarding in NDN networks," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2018, pp. 1–6.
[28] W. Drira and F. Filali, "A Pub/Sub extension to NDN for efficient data collection and dissemination in V2X networks," in *Proc. 15th IEEE Int. Symp. World Wireless, Mobile Multimedia Netw.*, Jun. 2014, pp. 1–7.
[29] J. Chen, M. Jahanian, and K. K. Ramakrishnan, "Black ice! Using information centric networks for timely vehicular safety information dissemination," in *Proc. IEEE Int. Symp. Local Metrop. Area Netw. (LANMAN)*, Jun. 2017, pp. 1–6.
[30] Z. Yan, S. Zeadally, and Y.-J. Park, "A novel vehicular information network architecture based on named data networking (NDN)," *IEEE Internet Things J.*, vol. 1, no. 6, pp. 525–532, Dec. 2014.
[31] R. Kleinberg, "Geographic routing using hyperbolic space," in *Proc. 26th IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2007, pp. 1902–1909.
[32] F. Papadopoulos, D. Krioukov, M. Boguna, and A. Vahdat, "Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.
[33] D. Krioukov, F. Papadopoulos, M. Kitsak, A. Vahdat, and M. Boguñá, "Hyperbolic geometry of complex networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 82, no. 3, Sep. 2010, Art. no. 036106.
[34] M. Boguñá, F. Papadopoulos, and D. Krioukov, "Sustaining the Internet with hyperbolic mapping," *Nature Commun.*, vol. 1, no. 1, pp. 1–8, Dec. 2010.
[35] A. Afanasyev, C. Yi, L. Wang, B. Zhang, and L. Zhang, "Scaling NDN routing: Old tale, new design," NDN Project Team, Gaithersburg, MD, USA, Tech. Rep. NDN-0004, 2013.
[36] R. Aldecoa and D. Krioukov. *Greedy Forwarding on the NDN Testbed*. [Online]. Available: https://www.caida.org/research/routing/greedy_forwarding_ndn/#HGCN10
[37] V. Lehman, A. Gawande, B. Zhang, L. Zhang, R. Aldecoa, D. Krioukov, and L. Wang, "An experimental investigation of hyperbolic routing with a smart forwarding plane in NDN," in *Proc. IEEE/ACM 24th Int. Symp. Qual. Service (IWQoS)*, Jun. 2016, pp. 1–10.
[38] S. Dulal and L. Wang, "Poster abstract: Experimental comparison between geohyperbolic and hyperbolic routing in NDN," in *Proc. IEEE Conf. Comput. Commun. Workshops*, Apr./May 2019, pp. 1045–1046.
[39] W. Yang, Y. Qin, Z. Yi, and Y. Yang, "Content-based hyperbolic routing and push mechanism in named data networking," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[40] Y. Wang, Z. Li, G. Tyson, S. Uhlig, and G. Xie, "Optimal cache allocation for content-centric networking," in *Proc. 21st IEEE Int. Conf. Netw. Protocols (ICNP)*, Oct. 2013, pp. 1–10.
[41] D. Rossi and G. Rossini, "On sizing CCN content stores by exploiting topological information," in *Proc. IEEE INFOCOM Workshops*, Mar. 2012, pp. 280–285.
[42] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proc. 2nd. Workshop Inf.-Centric Netw. (ICN)*, 2012, pp. 55–60.
[43] I. Psaras, W. K. Chai, and G. Pavlou, "In-network cache management and resource allocation for information-centric networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 11, pp. 2920–2931, Nov. 2014.
[44] A. Ioannou and S. Weber, "Towards on-path caching alternatives in information-centric networks," in *Proc. 39th Annu. IEEE Conf. Local Comput. Netw.*, Sep. 2014, pp. 362–365.
[45] A. Dabirmoghaddam, M. M. Barijough, and J. Garcia-Luna-Aceves, "Understanding optimal caching and opportunistic caching at 'the edge' of information-centric networks," in *Proc. 1st ACM Conf. Inf.-Centric Netw.*, 2014, pp. 47–56.
[46] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache 'less for more' in information-centric networks," *Comput. Commun.*, vol. 36, no. 7, pp. 758–770, 2013.
[47] K. C. Dey, A. Rayamajhi, M. Chowdhury, P. Bhavsar, and J. Martin, "Vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication in a heterogeneous wireless network—Performance evaluation," *Transp. Res. C, Emerg. Technol.*, vol. 68, pp. 168–184, Jul. 2016.
[48] M. H. Cheung, F. Hou, V. W. S. Wong, and J. Huang, "DORA: Dynamic optimal random access for vehicle-to-roadside communications," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 4, pp. 792–803, May 2012.
[49] X. Ma, J. Zhao, Y. Gong, and X. Sun, "Carrier sense multiple access with collision avoidance-aware connectivity quality of downlink broadcast in vehicular relay networks," *IET Microw., Antennas Propag.*, vol. 13, no. 8, pp. 1096–1103, Jul. 2019.
[50] J. Zhao, Y. Chen, and Y. Gong, "Study of connectivity probability of vehicle-to-vehicle and vehicle-to-infrastructure communication systems," in *Proc. IEEE 83rd Veh. Technol. Conf. (VTC Spring)*, May 2016, pp. 1–4.
[51] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.
[52] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 36–44, Jun. 2017.
[53] I. Ku, Y. Lu, M. Gerla, R. L. Gomes, F. Ongaro, and E. Cerqueira, "Towards software-defined VANET: Architecture and services," in *Proc. 13th Annu. Medit. Ad Hoc Netw. Workshop (MED-HOC-NET)*, Jun. 2014, pp. 103–110.
[54] E. Lee, E.-K. Lee, M. Gerla, and S. Oh, "Vehicular cloud networking: Architecture and design principles," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 148–155, Feb. 2014.
[55] C. Yi, J. Abraham, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "On the role of routing in named data networking," in *Proc. 1st Int. Conf. Inf.-centric Netw. (INC)*, 2014, pp. 27–36.
[56] A. Marandi, T. Braun, K. Salamatian, and N. Thomos, "BFR: A Bloom filter-based routing approach for information-centric networks," in *Proc. IFIP Netw. Conf. Workshops*, Jun. 2017, pp. 1–9.
[57] F. Papadopoulos, C. Psomas, and D. Krioukov, "Network mapping by replaying hyperbolic growth," *IEEE/ACM Trans. Netw.*, vol. 23, no. 1, pp. 198–211, Feb. 2015.
[58] *ndnSIM*. Accessed: 2020. [Online]. Available: http://ndnsim.net/current/
[59] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, and A. Rabuffi. (Jul. 2014). *CRAWDAD Dataset Roma/Taxi (V. 2014-07-17)*. [Online]. Available: https://crawdad.org/roma/taxi/20140717

NING YANG (Student Member, IEEE) received the master's degree in computer engineering from the University of Massachusetts Amherst, in 2006. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Southern Illinois University. Her research interests include future network architectures, network security, and connected vehicles.

KANG CHEN (Member, IEEE) received the B.S. degree in electronics and information engineering from the Huazhong University of Science and Technology, China, in 2005, the M.S. degree in communication and information systems from the Graduate University of Chinese Academy of Sciences, China, in 2008, and the Ph.D. degree in computer engineering from Clemson University, in 2014. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Southern Illinois University. His research interests include emerging wireless networks, datacenter networks, and software defined networking.



YAOQING LIU (Member, IEEE) received the master's and Ph.D. degrees from the University of Memphis in Computer Science. He is currently an Assistant Professor with the School of Computer Sciences and Engineering, Fairleigh Dickinson University. His publications appear in highly reputed conference proceedings and journals, such as the IEEE INFOCOM, ACM SIGCOMM CCR, and ACM/IEEE ANCS. His research interests include networked systems (security, routing, algorithm, and measurement) and named data networks (NDN). He has been a TPC Member and Technical Article Reviewer of the IEEE conferences, journal magazines, and Transactions.