Abstract—Data replication and in-network storage are two basic principles of the information-centric networking (ICN) framework in which caches spread out in the network can be used to store the most popular contents. This paper shows how one of the ICN architectures, i.e., Named Data Networking (NDN), with content prefetching can maximize the probability that a user retrieves the desired content in a vehicle-to-infrastructure scenario. We give an integer linear programming formulation of the problem of optimally distributing content in the network nodes while accounting for the available storage capacity and the available link capacity. The optimization framework is then leveraged to evaluate the impact on content retrievability of topology- and network-related parameters as the number and mobility models of moving users, the size of the content catalog, and the location of the available caches. Moreover, we show how the proposed model can be modified to find the minimum storage occupancy to achieve a given content retrievability level. The results obtained from the optimization model are finally validated against an NDN architecture through simulations in ndnSIM.

Index Terms—Content store, information-centric networking (ICN), integer linear programming (ILP), Named Data Networking (NDN), ndnSIM, vehicular network.

I. INTRODUCTION

Today’s Internet is focused on content retrieval instead of point-to-point communication. In addition, users are mostly mobile, continuously changing their location. One of the main problems of user mobility is the intermittent connectivity that causes loss of packets. An information-centric networking (ICN) framework is the solution to the previous problems. Indeed, communication is based on a request/response model where the focus is on the content. This is the basis of all the ICN frameworks, where the communication paradigm shifts from retrieving the content from a given location to retrieving the content from the network.

The user no longer cares where the packets come from, but rather wants them as soon as possible. Thus, an end node that wants a given content issues an INTEREST message with the name of the desired content. Then, the network is responsible for choosing the best node to fetch the content from by means of a DATA packet. In particular, in the Named Data Networking (NDN) paradigm, every node can be potentially the best source of a given content for a given user (content consumer). Indeed, the widespread use of caching into the network nodes is one of the foundations of the NDN protocol.

This paradigm blends well with specific issues of user mobility, such as intermittent connectivity and changing topologies. Let us consider what happens during a handover when a user is downloading a video from YouTube in a moving car. As the car moves, the access point (AP), to which the user is connected, is no longer available, and a new connection should be established seamlessly. Thus, handover procedures are used to keep track where the user is, that is, which mobile base station or AP she is currently attached, to reroute the video chunks to the proper access network device. Differently, in an NDN scenario, when a consumer moves from one location to another, a seamless handover is easily achieved by reissuing the interest messages from the new location for the subsequent video chunks, and the network can then deliver the content from the best source for the new location. This means that the DATA packet can be sent from whichever network node that has the content in its local cache.

We focus here on exploiting the NDN paradigm in the context of vehicular networks. In particular, we analyze the beneficial effects of prefetching contents at static network nodes on the performance of vehicular-to-infrastructure (V2I) communication paradigms. In our setting, vehicular users move along predefined paths and get in touch with several APs along their travel. The users entering the area served by the NDN network ask for a content from a content catalog. All content is segmented in chunks; thus, the vehicular user can retrieve different parts of the content from different APs. Content is successfully retrieved if the user moves out of the area with all the chunks of the content.

We define the problem of optimally prefetching the content chunks in the network nodes to maximize the probability of average content retrieval. We show that the problem can be formulated as an integer linear programming (ILP) problem, which can be extended/modified to encompass the case where the minimum retrieval probability (out of all the mobile users)
Conclusions are presented in Section VIII.

This paper is structured as follows. Section II provides an overall view of the related work. Section III reviews some of the basic concepts of the NDN architecture, which is taken as a reference in this work. Section IV explains the scenario considered in this paper and the behavior of the NDN-enabled nodes. The optimization models for content placement are presented in Section V. The numerical results are shown in Section VI. Conclusions are presented in Section VIII.

II. RELATED WORK

Mobile ICNs have recently attracted much attention within the research community. In [1] and [2], comprehensive surveys on ongoing research in mobile ICN are provided. The benefits that different ICN designs (e.g., data-oriented network architecture (DONA) [3], content-centric networking (CCN) [4], Network of Information (NetInf) [5], and NDN [6]) provide to content and producer mobility are highlighted in the aforementioned studies. In details, the first work mainly focuses on open challenges that should be addressed, whereas the second explains how to exploit the named-data paradigm for making the mobility management easier. In particular, in [1], the issues of producer mobility and dynamic routing are addressed, further considering the benefits in the management of user multihoming and handover. Differently, in [2], the performance improvement due to in-network caching and the adaptability to various scenarios are evaluated. While the aforementioned work is not scenario specific, the ICN paradigm is applied to specific vehicular ad hoc network (VANET) scenarios in [7] and [8]. The latter proposes a geo-based forwarding strategy and studies caching to avoid redundancy problem. Three heuristic strategies have been suggested and evaluated, showing an improvement in network performance. On the other hand, in [7], it is shown how the adoption of NDN could bring benefits to vehicular communications by implementing a prototype of VANET NDN.

In the field of vehicular networks, in [9], the question of whether CCN could be the solution for vehicular networks is raised. Evidence that content centric vehicular networking performs better than the legacy Transmission-Control-Protocol-based and/or Internet-Protocol-based architecture is provided. In addition, the advantages of using ICN for mobility support is also considered. In [10], a mobility management scheme for CCN that confirms the previous conjectures is proposed. It evaluates the routing update latency and the delivery latency depending on the number of nodes. The results show that the mobile CCN scheme achieves better performance than the basic CCN protocol. Nonetheless, the work does not evaluate how to optimize content distribution to lower the perceived latency and to guarantee content retrievability.

Several papers present solutions to optimize content distribution by exploiting content prefetching for both ICN and non-ICN scenarios. A proactive caching algorithm for NDN is proposed in [11], where the main idea is to proactively ask and cache contents before the user moves from one AP to another. The simulations show that the proposed approach has better performance than the original NDN: lower handover cost, higher delivery ratio, and shorter handover latency. The solution is implemented modifying the INTEREST packet and the basic communication protocol. Differently than our work, no optimization model is given for content prefetching, but rather, the proposed scheme is evaluated mainly through simulations.

The problem of optimizing content prefetching in “classical” network scenarios has been largely debated in the past years. As an example, in [12], the aim is to reduce the content retrieval in the World Wide Web by predicting and prefetching those files that are most likely to be requested. The prefetching is done based on server advices to the client, who can choose to prefetch or not the suggested files; the model makes probabilistic predictions for prefetching based on received requests.

A predictive prefetching method is also presented in [13] for Wi-Fi networks. New handoffs and data transfer strategies to reduce the connection setup latency and the download latency are developed. The results show that the performance of the vehicular WiFi network could be improved using the proposed strategies. However, some open challenges are left, for example, how to implement the estimation mechanism for prefetching. Our work does not consider the handoff strategy because the ICN context does not need to establish end-to-end connections. Both the previous papers do not present an optimization model for content prefetching, but show that prefetching can help in performance improvement for fixed and mobile scenarios, whereas our paper considers a new optimization strategy for content prefetching in ICN.

Then, in [14], the vehicular mobility is modeled using a fog-of-war probabilistic representation. The model is used to build a graph to formulate a non-ILP optimization problem. The problem is to decide the amount of traffic to offload to the vehicular network. The output of the problem is used to take the content prefetching decisions. A speculation-based prefetching scheme is proposed in [15] to ensure the users’ quality of experience. Moreover, it guarantees, together with a new grouping-based storage strategy, good data delivery efficiency and a high lookup success rate. On the other hand, an interesting mobility
similarity model is presented in [16] to enhance the data delivery efficiency. Users are grouped in a mobile community based on similar playback and movement. These three studies provide an overview of the current research trend for content prefetching in V2I network scenario.

Finally, in [17], the optimal caching policy for file servers colocated with the APs in a WiFi-based content distribution community infrastructure is analyzed. The content management problem is formulated as mixed-integer programming with the aim of maximizing the file retrieval probability within a time interval. The model considers 100 content files whose popularity follows Zipf’s law with different skew parameters. Moreover, it uses 50 APs with a coverage of 250 m. Our paper shares the same objective function of the aforementioned work, but on the other hand, our optimization framework considers additional constraints that are specific to a multihop NDN scenario.

III. BACKGROUND ON NAMED DATA NETWORKING

This section briefly reviews the core components of NDN [6], which constitute the reference networking paradigm for this work.

NDN Packet Model: The request is contained in an INTEREST packet, whereas the response is in a DATA packet. An INTEREST packet carries the Content Name, the Selector, and aNonce. The DATA packet is composed of the Content Name, the Signature, the Signed Info, and the Data. A DATA packet is the corresponding response of an INTEREST only if the Content Name in the INTEREST matches the prefix of the Content Name in the DATA packet. The names are typically hierarchical; thus, matching a prefix means that the Content Name in the DATA is in the subtree specified by the Content Name in the INTEREST.

NDN Node Model: Each node has three main data sets: the Content Store (CS), the Pending Interest Table (PIT), and the Forwarding Information Base (FIB). The CS is an associative container of data. Which data are stored in a node at a given time is decided by means of a CS management policy. The PIT stores the name prefixes that correspond to the INTEREST that the node could not satisfy and that it has sent to some other nodes; moreover, the node keeps track of the requesting nodes asking for the content, to send downstream the returned data. Finally, the FIB registers the prefixes and the corresponding list of neighbors to forward INTEREST packets.

NDN Communication Model: A node asks for a content by sending an INTEREST packet to all neighbors listed in the FIB for the matching prefix. After hearing the request, any node with the content responds with a DATA packet. When a node receives an INTEREST, it checks if there is a correspondence in its tables, i.e., CS, PIT, and FIB. If the CS caches the requested DATA packet, the node sends out the content and drops the satisfied INTEREST. Otherwise, if the match is in the PIT, the corresponding entry is updated adding the requesting node, and the INTEREST is discarded. If the match is in the FIB, the INTEREST is sent out to the next hop(s), and it is created a new entry in the PIT. Finally, if there is no match, the INTEREST is discarded because the node does not know how to find any matching DATA.

The DATA packet processing is quite similar; the node does a longest match lookup of the DATA packet Content Name; if there is a match in the CS, the node throws it away because it is a copy. Otherwise, the node looks in the PIT, and if there is a match, it sends the data to the requesting nodes. In case of a reactive CS policy, the node adds the packet to the CS, possibly dropping some other content. An FIB match means an unrequested DATA; thus, the node drops the packet.

IV. VEHICLE-TO-INFRASTRUCTURE SCENARIO FOR NAMED DATA NETWORKING

This paper considers a V2I scenario composed of a set of APs to provide connectivity to moving vehicles and a backbone network, which interconnects the APs (see Fig. 1).

The set of network nodes is identified by \( N \), where \( I \subseteq N \) is the set of APs. Each node in the network (vehicles, APs, backbone nodes) runs a simple version of the NDN protocol [6]. That is, when a vehicle enters the area and connects to the first AP along its path, it issues an INTEREST message for the first chunk of content \( j \) from the content catalog, which is composed of \( S_j \) chunks, each of size \( D(ck) \). Each AP has a Content Store, capable of holding CS chunks. If the AP receiving the INTEREST message has the requested chunk of the object \( j \), then it delivers it to the vehicle; otherwise, it issues an INTEREST message upstream to retrieve the requested chunks. The same procedure is repeated by any node in the backbone. If they have the requested chunks, they send them downstream; otherwise, they propagate the INTEREST upstream. The Producer node is a special node in the backbone, which is assumed to have all the chunks of all the contents and always satisfies the requests. A vehicle keeps sending INTEREST messages as long as it receives chunks or it moves outside of the AP coverage area. When the vehicle connects to a new AP, it starts resissuing requests for the missing chunks, until the content is fully received.

The reference scenario includes \( U_{tot} \) users moving at average speed \( s \), which may also change along the user’s path. The overall mobility pattern is modeled as follows: Each user is randomly assigned to one moving path out of \( V \) possible paths. The \( v \)th path has a probability \( \beta_v \) of being chosen that follows a Zipf’s law with exponent \( \alpha_p \), i.e.,

\[ \beta_v \sim \text{Zipf}(\alpha_p). \]
To evaluate the feasibility of such path-assignment model and to verify the previous assumption in a practical case, we have analyzed real-life mobility traces of a medium/large urban environment described in [18]. In particular, we have measured the path occurrence as a function of the path rank for the reference data set. As shown in Fig. 2, the probability to choose a path (path occurrence) obeys Zipf’s law, since the log–log plot of the path occurrence is a function in the path rank.

Each moving path then determines the set and sequence of APs visited by the moving user along its way. The APs are assumed to have a transmission diameter \( d \); consequently, the connection time available to one user at a given AP is defined as \( T_{\text{con}} = d/s \). For the sake of presentation, the time needed to discover and associate to the APs is not considered; however, the model/scenario can be trivially extended by scaling down the \( T_{\text{con}} \) parameter by a factor that depends on the discovery/association time.

The binary parameter \( m_{iv} \) represents if a user is connected with the AP \( i \) along the path \( v \). By letting \( m_{iv} \) be equal to 1 if a user along path \( v \) connects to AP \( i \), and 0 otherwise, the number of users on each path can be written as \( U_v = U_{\text{tot}} \cdot \beta_v \), whereas the average number of users connected to the \( i \)th AP per path \( v \) is

\[
U_i = \frac{\sum_{v=1}^{V} U_v m_{iv}}{\sum_{v=1}^{V} m_{iv}}.
\]

The content catalog is composed of \( C \) content objects. The \( j \)th content has a probability of being requested that follows the Zipf’s distribution with exponent \( \alpha_j \), as mostly assumed by the literature [19], [20]. The probability that the \( j \)th content is requested is \( \sigma_j \sim \text{Zipf}(\alpha_j) \).

Let us define the set of links in the reference network \( \mathcal{E} \); the available link bandwidth is \( c_e \), with \( e \in \mathcal{E} \). On the other hand, the radio channel has a bandwidth \( c_R \). We further assume a generic but known routing pattern in the reference scenario, that is, for every node in the network, it is known the set of edges (and ordered nodes) that constitute the shortest path to/from the reference node. In particular, it is defined the set \( \mathcal{E}_{ij} \), which includes the sequence of edges that belong to the shortest path from node \( i \) to node \( j \).

The maximum number of content chunks that can be transferred over link \( e \) downstream toward the \( i \)th AP for the users in the \( v \)th path is called the maximum downloadable burst (MDB), i.e., \( B_{evv} \), and depends on the available bandwidth, which must be shared among all the requests from the downstream nodes.

Assuming that the bandwidth of the link is equally shared by all the users connected to the same AP \( i \) along the path \( v \), we have

\[
B_{evv} = \frac{c_e \cdot T_{\text{evv}}}{D(ck)} \left( \sum_{i \in T_{evv} \in \mathcal{E}_{ij}} U_i \right)^{-1}
\]

where \( T_{\text{evv}} = T_{\text{con}} \cdot m_{iv} \) is the duration of the connection between AP \( i \) and the user along the path \( v \), \( T_{\text{con}} \) is the total duration of the connection between the user and the APs, and \( \mathcal{E}_{ij} \subset \mathcal{E} \) is the subset of links that belong to the shortest path toward AP \( i \). The function \( t(e) \) returns the tail of the edge \( e \). Thus, given an edge \( e \), which is the directed link from node \( a \) to \( b \), the function \( t(e) \) returns the node \( a \). On the other hand, the MDB associated to the radio channel, i.e., \( B_{ivv}^R \), toward the \( i \)th AP for users in the \( v \)th path is computed as follows:

\[
B_{ivv}^R = \frac{c_R \cdot T_{evv}}{D(ck)} \cdot U_i.
\]

It is also worth noting that retrieving chunks from a backbone node “closer” to the Producer node incurs in a transmission and processing overhead. The parameter \( \eta_{evv} \geq 1 \) represents the ratio between the time needed to retrieve at AP \( i \) a content chunk for a user along path \( v \) through a backbone link \( e \) and the time to retrieve the same chunk if the Content Store is available at AP \( i \). This factor can be computed as

\[
\eta_{evv} = \frac{\sum_{f \in \text{SP}(c, e)} T_{(ck)} f + 2 \tau \cdot |\text{SP}(c, e)|}{T_{evv} (ck) R + 2 \tau} + 1
\]

where \( T_{evv} (ck) = D(ck) / B_{evv} \) is the transmission time for a single chunk using the available MDB through link \( e \), and \( T_{(ck)}^R = D(ck) / B_{ivv}^R \) is the corresponding MDB through the radio channel. A transmission latency \( \tau \) accounts for the processing cost of the messages.

V. OPTIMAL CONTENT PLACEMENT

This section provides the ILP formulation for the problem of content placement. Differently than the reference NDN protocol, we make the following assumptions.

1) The chunks in the Content Stores of the network nodes do not change over time according to a caching policy, but are prefetched according to the decisions of an offline management platform.

2) The content objects are protected with a forward error correction code, which encodes a file of size \( S_j \) chunks in \( H \) chunks. The \( j \)th content can be fully reconstructed if the consumer obtains any \( S_j \) chunks. Therefore, the Consumer does not issue INTEREST messages for specific chunks but issues general INTERESTs for additional chunks.

A. Maximizing the Content Retrieval

The optimization objective is to distribute content chunks into the Content Stores to maximize their availability for the
For the sake of presentation, we assume in the following that the following sets.

- Access Points: i ∈ I = {1, ..., I}
- Contents: j ∈ J = {1, ..., C}
- Content Stores: k ∈ K = {1, ..., K}
- Paths: v ∈ V = {1, ..., V}
- Links: e ∈ E = {1, ..., E}
- Shortest Paths: s ∈ SP_{ij} = {1, ..., N × N}

For the sake of presentation, we assume in the following that N = K, that is, all the nodes in the network (APs and backbone nodes) can cache contents. Moreover, we further assume that the content Producer is assigned index 0.

The proposed formulation leverages the decision variables and parameters resumed in Table I.

The problem of prefetching content at network content stores such that the probability of retrieving contents is maximized can be formalized as follows:

$$\max : \sum_{j \in J} \sigma_j \sum_{v \in V} \beta_v A_{jv} \tag{3}$$

s.t.

$$\sum_{i \in I} z_{ivj} \geq A_{jv} S_j \quad \forall j \in J \forall v \in V \tag{4}$$

$$\sum_{i \in I} z_{ivj} \geq \sum_{j \in J} y_{eivj} m_{iv} \geq A_{jv} S_j \quad \forall j \in J \forall v \in V \tag{5}$$

$$\sum_{f \in \text{SP}_{iv}} y_{eivj} \leq B_{iv} \quad \forall i \in I \forall v \in V \forall e \in E \forall j \in J \tag{6}$$

$$z_{ivj} \leq x_{ji} \quad \forall i \in I \forall v \in V \forall j \in J \tag{8}$$

$$\sum_{j} x_{jk} \leq C_{S_k} \quad \forall k \in K \tag{9}$$

$$y_{eivj} = 0 \quad \forall i \in I \forall v \in V \forall j \in J \forall e \in E : e \notin \text{SP}_{iv} \tag{10}$$

$$x_{jk} \geq 0, \quad y_{eivj} \geq 0, \quad z_{ivj} \geq 0, \quad A_{jv} \in \{0, 1\}. \tag{11}$$

The objective is to maximize the retrievability of the content j, i.e., A_{jv}, which is the probability of satisfying the request of the vehicular user. The objective function (3) depends on the probability of requesting the content j, i.e., σ_j, and on the probability of choosing the path v, i.e., β_v. The first constraints (4) define the retrievability of a content. A content is retrieved if the overall number of chunks that are retrieved from any network node (AP and backbone node) is above the required threshold. Constraints (5) and (6) impose a limit on retrievability that depends on the MDB. In short, the overall number of chunks retrieved from network nodes closer to the Producer cannot exceed the capacity of the network links used to deliver them downward. The parameter y_{eivj} emphasizes the fact that the farther the chunks are in the network, the higher the cost to retrieve them will be. The number of chunks cached in each content store constrains the maximum number of retrievable chunks, as enforced by constraints (7) and (8). Equation (9) introduces budget-type constraints on the maximum number of contents/chunks stored at any network node. Then, (10) does not retrieve chunks from links that are not comprised in the set of the shortest paths between the considered AP and the producer node. Finally, (11) defines the decision variables of the formulation.

**B. Maximizing the Worst Content Retrievability**

The model introduced in the previous section maximizes the average content retrievability; thus, fairness among different paths/users may be low. In this context, it is also worth enforcing a more fair solution by maximizing the “worst,” i.e., the minimum, content retrievability out of all the mobile users. We introduce a new variable, i.e., ϵ, that we would like to maximize in our objective function. The value of this variable is determined from the first set of constraints (13), where the parameter β_v has been removed with respect to (3) in the original formulation to guarantee a fair comparison among the paths. This allows us to study the impact of caching on paths with different popularity.

The objective function is as follows:

$$\max : \epsilon. \tag{12}$$

The constraints are as follows:

$$\sum_{j} \sigma_j A_{jv} \geq \epsilon \quad \forall v \in V \tag{13}$$

$$x_{jk} \geq 0, \quad y_{eivj} \geq 0, \quad z_{ivj} \geq 0, \quad A_{jv} \in \{0, 1\}, \epsilon \in [0, 1]. \tag{14}$$

The constraints from (4) to (10) are unchanged. The variables and the indices remain the same, as in Section V-A.

The objective function (12) maximizes ϵ, which represents a probability; indeed, ϵ is the smallest success probability.
within the success probability associated to the different paths \( v \) in the network, which is deduced from the constraints (13). Equation (14) adds the information that \( \epsilon \) should be within 0 and 1.

C. Minimizing the Total Size of Content Stores

The previous two formulations target the maximization of the content retrievability metric under a given budget in terms of available Content Store size. Here, our aim is to minimize the total size of the content store to guarantee a desired success probability. We introduce a new parameter that is the minimum content retrievability \( P_{\text{succ}} \).

The objective function is as follows:

\[
\min : \sum_{j,k} x_{jk}.
\]

(15)

The constraints are as follows:

\[
\sum_{j,v} \sigma_{jv} \beta_v A_{jv} \geq P_{\text{succ}}.
\]

(16)

The constraints from (4) to (11) are unchanged, except for constraint (9), which is deleted. The variables and the indices remain the same, as in Section V-A.

The objective function (15) minimizes the sum of all the \( x_{jk} \) that represents the number of chunks stored in the network content stores. The new constraint (16) imposes a lower bound on the content retrievability.

D. On the Problem Size

For the sake of completeness, we provide the asymptotic complexity of the problem formulation expressed as a function of the number of variables and constraints. The complexity is similar for all the three models. The number of variables of the first model Section V-A is

\[
O(C \cdot I \cdot V \cdot E + C \cdot K).
\]

On the other hand, the number of constraints is

\[
O(K + I \cdot V \cdot C \cdot E).
\]

The second model V-B has an additional variable and \( O(V) \) additional constraints. Finally, the third model Section V-C has the same number of variables and \( O(K) \) fewer constraints.

In practical network scenarios, the largest term in the preceding expression is likely the number of possible paths followed by the vehicular users. To this extent, we discuss in Section VI-D how the number of paths actually used does not grow fast with the size of the reference mobility arena, thus extending the usability of the proposed model formulation to real-life environments.

VI. Performance Evaluation

Here, we show the results obtained by solving the ILP model by means of A Mathematical Programming Language (AMPL) with the solver IBM ILOG CPLEX Optimization Studio. We start off by focusing on a tree-structured network topology, as this is one of the most widespread solutions for the backbone implementation (see Fig. 3). Routing to/from the APs happens along a shortest path tree as represented in the figure. Later on here, we further apply our optimization problem to more general network topologies, namely, circle and meshed topologies (see Figs. 4 and 5, respectively). Table II further summarizes the scenario parameters and the values used in the optimization,
TABLE II
SCENARIO PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau$</td>
<td>link latency</td>
<td>0.25 ms</td>
</tr>
<tr>
<td>$c^R$</td>
<td>radio channel capacity</td>
<td>9 Mbit/s</td>
</tr>
<tr>
<td>$c_e$</td>
<td>backbone link capacity</td>
<td>1 Gbit/s</td>
</tr>
<tr>
<td>$L$</td>
<td>tree depth</td>
<td>4</td>
</tr>
<tr>
<td>$E$</td>
<td>number of links</td>
<td>28/30/36</td>
</tr>
<tr>
<td>$I$</td>
<td>number of access points</td>
<td>8</td>
</tr>
<tr>
<td>$K$</td>
<td>number of content stores</td>
<td>14</td>
</tr>
<tr>
<td>$C$</td>
<td>size of the content catalog</td>
<td>50</td>
</tr>
<tr>
<td>$\sigma_r$</td>
<td>exponent of content popularity</td>
<td>1</td>
</tr>
<tr>
<td>$V$</td>
<td>number of paths</td>
<td>5</td>
</tr>
<tr>
<td>$\sigma_p$</td>
<td>exponent of path popularity</td>
<td>1</td>
</tr>
<tr>
<td>$U_{tot}$</td>
<td>total number of users</td>
<td>700</td>
</tr>
<tr>
<td>$S_j$</td>
<td>content size</td>
<td>1000 chunk</td>
</tr>
<tr>
<td>$D^{(ch)}$</td>
<td>chunk size</td>
<td>1000 byte</td>
</tr>
<tr>
<td>$CS_k$</td>
<td>content store size</td>
<td>2000 chunk</td>
</tr>
<tr>
<td>$d$</td>
<td>diameter of AP coverage</td>
<td>250 m</td>
</tr>
<tr>
<td>$s$</td>
<td>vehicle speed</td>
<td>90 km/h</td>
</tr>
<tr>
<td>$T_{con}$</td>
<td>total duration of connection</td>
<td>10 s</td>
</tr>
<tr>
<td>$m_{nu}$</td>
<td>connection indicator</td>
<td>1</td>
</tr>
</tbody>
</table>

Fig. 6. Success probability $P_{succ}$ in content retrieval depending on Content Store size $CS$, assuming storage at different levels of the network.

The network costs depend on the content store size, the available bandwidth, and the number of APs per path. To study how these design choices impact on the success probability under different network and mobility conditions, we use three slightly different models to expose different points of view. A model striking a balance among the different objectives according to some policy can be easily obtained.

A. Maximizing the Content Retrievability

We evaluate the importance that storage has at the different levels of the tree. Fig. 6 shows the success probability in content retrieval depending on Content Store size assuming that storage is available only at the APs, at the APs plus the second level of nodes, or at all the levels. As the Content Store size increases, the success probability also increases. Having storage only at the APs however does not allow the network to achieve its full potential. Adding storage to the nodes of the second level increases the success probability. However, as stated in [22], the performance improvement is at most 17% relative to the first level. Adding storage to the third level nodes as well slightly increases the success probability, but only to a limited extent, which becomes null as the storage in the outer nodes grows.

Fig. 7 shows the success probability $P_{succ}$ in content retrieval depending on contact time ratio $CTR$, assuming storage at different levels of the network.

The results reported in Figs. 6 and 7 lead us to the same conclusions in [22]. Indeed, by using a sensitivity analysis, in [22], it is shown that an NDN architecture with pervasive caching and nearest replica routing can provide at most 17% of best case improvement in network performance over a simple edge-based caching architecture. Moreover, it is proven that if the edge caches are doubled, an edge-caching architecture performs even better than NDN. Thus, as concluded in [22], we can state that making pervasive use of content stores does not provide substantial advantages that could justify to add additional complexity to the network. On the other hand, we can think that the gain provided by the additional level of content stores is at most 17%, relative to the first level.

The results reported in Figs. 6 and 7 lead us to the same conclusions in [22]. Indeed, by using a sensitivity analysis, in [22], it is shown that an NDN architecture with pervasive caching and nearest replica routing can provide at most 17% of best case improvement in network performance over a simple edge-based caching architecture. Moreover, it is proven that if the edge caches are doubled, an edge-caching architecture performs even better than NDN. Thus, as concluded in [22], we can state that making pervasive use of content stores does not provide substantial advantages that could justify to add additional complexity to the network. On the other hand, we can think that the gain provided by the additional level of content stores is at most 17%, relative to the first level.

The results reported in Figs. 6 and 7 lead us to the same conclusions in [22]. Indeed, by using a sensitivity analysis, in [22], it is shown that an NDN architecture with pervasive caching and nearest replica routing can provide at most 17% of best case improvement in network performance over a simple edge-based caching architecture. Moreover, it is proven that if the edge caches are doubled, an edge-caching architecture performs even better than NDN. Thus, as concluded in [22], we can state that making pervasive use of content stores does not provide substantial advantages that could justify to add additional complexity to the network. On the other hand, we can think that the gain provided by the additional level of content stores is at most 17%, relative to the first level.

The results reported in Figs. 6 and 7 lead us to the same conclusions in [22]. Indeed, by using a sensitivity analysis, in [22], it is shown that an NDN architecture with pervasive caching and nearest replica routing can provide at most 17% of best case improvement in network performance over a simple edge-based caching architecture. Moreover, it is proven that if the edge caches are doubled, an edge-caching architecture performs even better than NDN. Thus, as concluded in [22], we can state that making pervasive use of content stores does not provide substantial advantages that could justify to add additional complexity to the network. On the other hand, we can think that the gain provided by the additional level of content stores is at most 17%, relative to the first level.
Fig. 8. Success probability $P_{\text{succ}}$ in content retrieval, depending on content catalog size $C$, assuming different storage sizes.

store 2000 chunks. The smaller is the content store size and the bigger is the catalog, the smaller is the success probability.

The results reported so far show that large Content Stores, high link capacity, nonintermittent connections, and small content catalog all increase the success probability, but the effect is different depending on the other system parameters. In particular, increasing the wireless capacity and avoiding intermittent connections have a beneficial impact, but, at the same time, are unlikely to be controlled and changed. The size of the Content Store, on the other hand, is likely to be effortless to improve and can partly compensate a less dense network or a network with less capacity. However, we should take into account that spreading the content stores over a big network is far from being an easy task. Finally, we should consider that the bigger is the variety of content, the more important the size of the content stores becomes.

Fig. 9 shows the number of users in the network versus their speed. The continuous line represents the upper bound for the tradeoff between the number of users in the network and their speed to get a success probability equal to 1, assuming that all the contents chunks are stored in all the content stores of the first level.

The values are computed using the following equation:

$$U_{\text{tot}} \leq e^{R \cdot d \cdot I \cdot \sum_{i=1}^{I} m_{i1}} S_1 \cdot D(ck) \cdot s$$

where $s$ and $U_{\text{tot}}$ are the changing variables to get the chosen success probability. Note that $m_{i1}$ represents the number of contacts with the APs along the first path. The value is the same for all the paths.

The dots in the figure represent the number of users versus their speed to achieve 100% success probability, if the solution given by our model is exploited. We depict values varying the content store sizes $CS_k$, which are the same in all the tree levels. The bigger are the content stores, the nearer are the values obtained from our model to the bound value.

Thus, this figure highlights that the results provided by our model are comparable to the optimal bound. Moreover, also small content stores can reach 100% of success probability paying only a small price in terms of decreased speed and diminished number of users.

Then, we also consider the two more general topologies depicted in Figs. 4 and 5 with the same number of APs, backbone nodes, and one content Producer. Routing to/from APs along the shortest path is precomputed with respect to the topology.

Fig. 10 compares the success probability values under different backbone network topologies. In particular, the chart reports the success probability $P_{\text{succ}}$ as a function of the content store size $CS$ and parametrized with respect to link latency $\tau$ (and, consequently, to $\eta_{eiv}$). To make the comparison fair among the different topologies, the circle and meshed topologies share the same number of APs as the tree topology and the same routing paradigm, that is, routes to/from APs adhere to the shortest path paradigm. The figure highlights the influence of the overall network latency, showing that the success probability is larger with smaller $\tau$. Therefore, if the links have a larger latency, the storage size must be increased to achieve the same success probability. Moreover, as can be expected, the storage size has a big effect: The bigger the content stores, the higher the success probability. It can be noticed that the tree topology provides a higher success probability than the circle and mesh topologies with small $\tau$. On the other hand, with bigger values
The second model tries to maximize the worst success probability among the possible paths. Thus, \( \epsilon \) represents this success probability in content retrieval relative to the path with the smaller probability of being chosen. We assume that there are 480 users that connect with only six over the eight possible APs. Fig. 11 represents the values of \( \epsilon \) as a function of the contact time ratio assuming storage at different levels of the network. It can be noticed that the success probability diminishes if the users move faster. Moreover, differently from Fig. 7, the values of content retrievability are very close for all the three considered situations; this is due to the new \textit{minmax} model that tries to balance the success probability of all the paths. Thus, we can say that adding levels of caching does not provide advantages in content retrievability among different paths. However, we have to pay the price of decreased performance.

C. Minimizing the Total Size of Content Stores

The third proposed model allows us to evaluate the size of the content stores to install in the network to guarantee a chosen success probability, which we fixed at \( P_{\text{succ}} = 0.95 \). The total content store size in the network is represented as \( \mathcal{C}_t \). We study the number of users that our network with different content store sizes can serve in Fig. 12. Moreover, we assume that content stores are available at different levels of the tree. We note that the content stores are useless with less than 660 users because the network can afford the load and retrieve the contents from the repository at the root of the tree. On the other hand, ranging from 660 to 720 users, it is possible to raise the content store sizes to achieve the fixed success probability. Then, if the users are more than 720, the network cannot supply the requests due to lack of bandwidth. Thus, content stores cannot overcome the limit imposed by the link access availability. Considering the case with content stores in all the tree levels, we noticed that, as the number of users grows, the content stores are pushed down toward the access network. On the contrary, the content stores in the upper levels are more effective when there are fewer users.

Fig. 13 shows the total content store size as a function of the exponent of the path popularity to achieve a success probability of 95% in content retrieval. We assume that there are 480 users that connect with only six out of the eight possible APs. It can be noticed that, when the users’ distribution across the available paths is more skewed, that is, when the congestion level of the more congested path increases, the size of the deployed content stores increases. Moreover, as it could be expected, if only the first level in the topology is available, it is necessary to install bigger content stores than the cases with also the second and third levels. This figure also confirms that the third level of content stores is less useful for the scenarios considered.

Fig. 14 represents the total content store size as a function of the chosen success probability in content retrieval. As it can be expected, the higher is the desired success probability, the bigger is the size of the content stores. Furthermore, moving from 80% of success probability to 100%, we should more than double the total content store capacity. Thus, if we would like to guarantee a higher probability in content retrieval, we should evaluate whether to invest more in content store is worth the costs. Moreover, it can be noticed that, by installing content stores not only in the APs, it is possible to reduce the total content store size and thus to save in costs.
Finally, Fig. 15 depicts the total content store size needed to achieve a chosen success probability, considering variable users’ speed. The sequence of long and short connections with the APs leads to installing bigger content stores in the network. However, the investment for the storage to guarantee the same chosen success probability in case of different contact time ratios is reasonable and provides an infrastructure that is more suitable for heterogeneous scenarios.

D. Discussion on the Usability of the Proposed Approach

As already observed in Section V-D, the cardinality of the set of possible paths $V$ may increase exponentially in real-life mobility environments, which would make the solution of the optimization problem impractical, or, in the best case, too slow. To assess the scalability of the proposed approach, we have considered real-life vehicular traces of the city of Zurich [18], which provide detailed information on users’ mobility in an area of 18.4 km$^2$; leveraging these traces, we have evaluated the total number of distinct paths that can be observed in such vehicular realization when varying the size of the observation area. In particular, Fig. 16 reports the total number of paths and the number of the most popular paths as a function of the size of the geographical region of observation (up to the complete Zurich area). Interestingly, the total number of paths and the number of most popular 50% paths are not exponential in the size of the area but, rather, feature a linear growth.

Fig. 16 shows the success probability when applying the optimization model considering a growing number of paths in the model parameter space starting from the most popular ones until all the paths are considered in the model. It is worth noting that, by considering only the number of paths that cover 50% of probability, i.e., about 20 paths, instead of the total number of paths, i.e., 330, the quality of the achieved solution is within 15% with respect to the optimum. Fig. 18 depicts the time to solve the problems with a growing number of paths. We can
users in the network using Little’s law, i.e., 
\[ U \] traffic pattern.
and managed in reactive ways, depending on the observed NDN network/scenario, in the dynamic case, the caches are filled in number of entering users per unit time, and
\[ U \] is the mean number of users in the system, against dynamic caching policie s. To this extent, we leverage model, which returns the optimal content prefetching strategy the caching strategy given
the ndnSIM simulator \[ [23] \], which implements an LRU caching
use to plan the content placement in realistic network scenarios.

It can be noticed that the success probability \( P_{\text{succ}} \) is 1 with less than 200 users and is 0 with more than 800 users in both cases. On the other hand, the gain in using the optimization model instead of the classical LRU policy ranges from 50% with about 400 users to a really big gain when the users are 700. Thus, we can say that the proactive placement of content into the content stores can provide big advantages over a reactive caching policy.

It is worth pointing out that proactive placement requires to have full knowledge of the content popularity and the users’ mobility patterns. To this extent, the observed gain of the proactive content placement with respect to dynamic cache management can be interpreted as an upper bound of the actual gain; on the other hand, there is evidence in the literature that content popularity is fairly stable over time. As an example, in [24], it is shown that near-future popularity can be predicted with relatively high accuracy (close to 90%), whereas popularity after 90 days can be predicted with an accuracy around 85%; this results support the fact that the gap between proactive content placement and dynamic cache management observed in our simulation is indeed representative of realistic network conditions.

**VIII. Conclusion**

This paper has exploited the in-network memory foundation of the NDN framework to simplify the delivery of the contents in a vehicular network. There are two critical aspects for the success of NDN: content placement and caching policy. Our work provides a modeling framework that is a starting point for solving these issues. We define a model that aims to maximize the probability that a vehicular node can obtain the requested content during its stay in the network. We evaluate the system in terms of success probability in content retrieval and investment on storage capacity in the network. By assuming that the Content Stores of the network nodes can be populated in advance, we provide an ILP formulation of the problem of optimally placing the content chunks in the network. The proposed optimization framework is then applied to realistic NDN scenarios to assess the impact of several network parameters onto the content retrievability. Such analysis provides insightful views on where to place content stores and which size of content store to place in ICNs. The following general guidelines/outcomes are sample results of the analysis carried out in this work.

- Increasing the storage capacity at the nodes can improve the success probability, particularly when the APs are sparse, when the content catalog is bigger, or when there are a lot of users in the network.
- The maximum gain is achieved by investing in storage capacity in the APs, that is, in the access network closer to the end users.
- Investing in storage capacity in the second level of network nodes also improves the performance of about 17%, particularly if the link latency is low.
- The access link capacity is often the bottleneck on content retrievability, even if infinite content store size is available.
• The mobility pattern of the users has an impact on the content store allocation; that is, when users move with heterogeneous speed values, in general, bigger content stores are needed and adding levels of storage helps in performance improvement.

REFERENCES


Giulia Mauri received the Bachelor’s degree in telecommunication engineering from Politecnico di Milano, Milan, Italy, in July 2010; the M.Sc. degree (cum laude) in telecommunication engineering also from Politecnico di Milano, in October 2012; and the Ph.D. (high qualified research doctor) degree in information and communication technology from Scuola Interpolitecnica di Dottorato (a joint project among Politecnico di Milano, Torino, and Bari) in April 2016.

Her undergraduate thesis was about the analysis of SAS data, her graduate thesis was about preserving privacy in a smart grid scenario, and her postgraduate thesis, which is entitled “Analysis of security issues in information centric networking,” covered problems related to trust management and privacy. In 2014, she was a Visiting Scholar with the University of California, Los Angeles, CA, USA. She is currently a Network Architect in the Network Services area of SIA S.p.A., Milan.

Her second price from the Italian Association for Information Security (CLUSIT) for her graduate studies thesis.

Mario Gerla (F’02) received the engineering degree from the Politecnico di Milano, Milan, Italy, and the Ph.D. degree from the University of California at Los Angeles (UCLA), Los Angeles, CA, USA.

He is currently a Professor of computer science with UCLA. During his Ph.D. program at UCLA, he worked on early Advanced Research Projects Agency Network (ARPANET) protocols under Prof. L. Kleinrock. Since joining the UCLA Faculty in 1976, he has contributed to several network designs, including wireless (ODMRP and CODECast) and Internet transport (TCP Westwood). He has led the ONR MINUTEMAN project (2001–2006), designing the next-generation scalable airborne Internet. At UCLA, he has developed a vehicular testbed for safe navigation and intelligent transport.

Prof. Gerla received the MILCOM Award (2011), the IEEE Ad Hoc and Sensor Network Award (2011), and the Sigmobile Life Achievement Award (2015).

Federico Bruno received the M.Sc. degree in telecommunication engineering from Politecnico di Milano, Milan, Italy, in 2014.

Since then, he has been with ICT Consulting S.p.A., Milan, where he is currently a Telecommunication Consultant.
Matteo Cesana (M’00) received the M.S. degree in telecommunications engineering and the Ph.D. degree in information engineering from Politecnico di Milano, Milan, Italy, in July 2000 and September 2004, respectively.

From September 2002 to March 2003, he was a Visiting Researcher with the Computer Science Department, University of California, Los Angeles, CA, USA. He is currently an Associate Professor with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano. His research activities are in the field of design, optimization, and performance evaluation of wireless networks, with a specific focus on communication technologies of the Internet of Things and future generation cellular networks.

Dr. Cesana is an Associate Editor of the AdHoc Networks journal (Elsevier).

Giacomo Verticale (S’00–M’03) received the Ph.D. degree from Politecnico di Milano, Milan, Italy, in 2003, defending a dissertation on the performance of packet transmission in third-generation mobile networks.

He is currently a Researcher with Politecnico di Milano, where he is also the Cohead of the Broadband Optical Networks, Security, and Advanced Internet (BONSAI) Laboratory of the Department of Electronics, Information, and Bioengineering (DEIB). Before joining Politecnico di Milano, he was with the CEFRIEL research center. His research interests are in network security and in performance evaluation of network protocols.