A QoS-aware Cache Replacement Policy for Vehicular Named Data Networks

Hakima Khelifi, Senlin Luo, Boubakr Nour, Hassine Moungla

A QoS-aware Cache Replacement Policy for Vehicular Named Data Networks

Hakima Khelifi[*], Senlin Luo[*], Boubakr Nour[‡], and Hassine Moungla[§][¶]
[*]School of Information and Electronics, Beijing Institute of Technology, Beijing, China
[‡]School of Computer Science, Beijing Institute of Technology, Beijing, China
[§]LIPADE, Paris Descartes University and Sorbonne Paris Cite University, France
[¶]CNRS, UMR 5157, Mines Telecom Institute, Telecom SudParis, CEA Nano-Innov Saclay, France
Emails: {hakima, luosenlin, n.boubakr}@bit.edu.cn, hassine.moungla@parisdescartes.fr

Abstract—**Vehicular Named Data Network (VNDN) uses Named Data Network (NDN) as a communication enabler. The communication is achieved using the content name instead of the host address. NDN integrates content caching at the network level rather than the application level. Hence, the network becomes aware of content caching and delivering. The content caching is a fundamental element in VNDN communication. However, due to the limitations of the cache store, only the most used content should be cached while the less used should be evicted. Traditional caching replacement policies may not work efficiently in VNDN due to the large and diverse exchanged content. To solve this issue, we propose an efficient cache replacement policy that takes the quality of service into consideration. The idea consists of classifying the traffic into different classes, and split the cache store into a set of sub-cache stores according to the defined traffic classes with different storage capacities according to the network requirements. Each content is assigned a popularity-density value that balances the content popularity with its size. Content with the highest popularity-density value is cached while the lowest is evicted. Simulation results prove the efficiency of the proposed solution to enhance the overall network quality of service.**

I. Introduction

Vehicular Named Data Network (VNDN) [1] aims to integrate the next-generation Internet architecture (i.e. Named Data Networking (NDN) [2]) on top of vehicular network. The use of NDN on top of vehicles networks has several advantages such as: simplifying the communication by using content names instead of host addresses [3], improving the security and privacy by using content based security [4], [5], enhancing the mobility support, and improving the content distribution and availability by using in-network caching.

In-network caching [6], in particular, plays an important role in content dissemination and delivery in VNDN. Any node may cache and deliver the content regardless of the original producer availability. Hence, the network becomes aware on content delivery; and therefore, improve the content availability, reduce the overall network load, eliminate single point of failure, and decrease the network delay. The caching in NDN may involve two different aspects: cache placement strategy [7], and cache replacement policy [8]. The former deals with where to cache the content, the latter deals what which content to cache and which content to remove in case the cache store is full.

In this context, several solutions have been proposed, most of them have been coupled with the forwarding plane. However, vehicular communication is characterized by the huge, dynamic, and diverse contents and traffic. Some of contents are popular and used multiple time in a long period of time (e.g., video streaming), others are less popular and may be used in a specific time (e.g., news), while others have a small period of usage and may be used only once (e.g., warning messages). The content caching must deal with this different type of traffic and improve the network/user quality of service (QoS).

These issues are the main motivation behind this work. Indeed, we propose a QoS-aware Cache Replacement (QCR) policy for vehicular networks. The proposed solution aims to cache not only the popular content, but the content with higher popularity-density from the cache store perspective (both content popularity and size are taken into consideration). Hence, improving the QoS parameters such as the network delay, cache utilization, and caching operations. Toward this, we classify the traffic into different classes, and virtually split the cache store into a set of sub-cache stores according to traffic classification. We formulate the cache placement problem as a Knapsack problem, where each sub-cache store has a predefined capacity according to the used scenario and caches only content related to its class.

QCR can scale without affecting the network performance. Indeed, by splitting the cache store and assigning to each content a popularity-density value, we can efficiency distribute and cache data over the network. For example, the near cache store to a group of consumers caches the most popular content according to its defined classes. Other cache stores that are far away may cache content with less popularity and so one. Hence, we can ensure that not only the popular content is cached near to consumers but also less-popular content is cached at the core network with trade-off between popularity, content size, and network delay. Another scenario can cache the critical information in a sub-cache store with large caching capacity at the edge node, and other traffic at the second level of cache stores. Thus, the dissemination of critical information can be guaranteed much faster than downloading a video or map, which may help to prevent attacks in the road. On the other hand, QCR enhances the impact of mobility by caching only the important contents regardless of producer or consumer mobility, the splitting feature also improves the cache diversity and distribution by minimizing the impact of re-issuing lost requests during the mobility.

II. Background & Related Work

This section provides background about NDN forwarding plane and review of existing caching solutions emphasizing on quality of service support.

A. NDN Forwarding Plane

Before diving into NDN caching, it is very important to understand the forwarding plane in NDN and how the caching is involved. The NDN forwarding plane is divided into two phases, as illustrated in Figure 1: (i) Interest Forwarding: consists of forwarding the requests in form of Interest packet upstream. When an intermediate node receives an Interest packet, it checks the Cache Store (CS) if the data is locally cached, if so, it forwards the data directly to the same received interface and drops the Interest packet. Otherwise, it checks the Pending Interest Table (PIT) if a similar request has been already forwarded, if so, it appends the received interface ID to the PIT entry and drops the Interest packet. Otherwise, it checks the Forwarding Information Base (FIB) table to find the forwarding interface; and (ii) Data Forwarding: consists of forwarding the content in form of Data packet downstream. After receiving a Data packet, the node checks its PIT if the request has been forwarded via the node or not, if not the packet will be dropped immediately, otherwise it is forwarded to all interfaces listed in PIT table, at the same time, the caching plane decides to cache the content in CS or not.

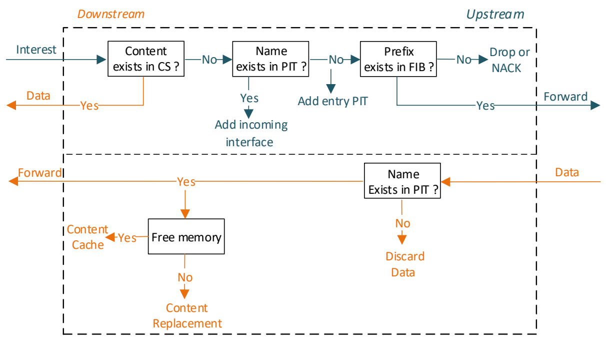


Fig. 1: NDN Forwarding Plane.

B. NDN In-Network Caching

In-network caching is a fundamental element in NDN that aims to cache content at different places in the network without involving any application layer decisions. Thus, the content is available at any-time from any-place, which may reduce the network load and latency. We distinguish two caching aspects [9]:

Cache Placement Strategy: This strategy tends to decide where to cache the content, or if the content should be cached at the node level or not. The most cache placement strategies in NDN are: Leave Copy Everywhere (LCE), Leave Copy Down (LCD), Edge Caching (EC), and Consumer Cache (CC) [10]. LCE places the content at every node in the communication path between the consumer and the producer, hence increases the content availability, but causes a lot of redundancy. LCD stores the content in one node after the producer. EC places the content in consumers' edge node, and CC caches the content in the node that is directly connected to the consumers.

Cache Replacement Policy: This policy decides which content should be evicted from the cache store to cache the new content if the CS is full. Most of the existing policies in NDN are: Least Recently Used (LRU), Least Frequently Used (LFU), First In First Out (FIFO), Random Replacement (RR), and Time-to-Live (TTL) [6]. LRU evicts the content that is rarely requested and prefers the most frequently one to cache. LFU evicts the content with the lowest usage frequency and selects the content with the highest usage frequency to cache. FIFO replaces the oldest content with the newly arrived one. RR removes a random content from the cache store and replaces it with the arrived one. Finally, TTL assigns to each cached content a timestamp, if a content reaches a predefined threshold, it will be removed from the CS.

C. Content Caching in NDN-based VANET

Existing caching placement schemes in VNDN can be grouped into: (a) *Probabilistic-based Caching* [11], [12], which is based on probabilistic estimation in order to permit nodes to decide whether to cache content or not in order to improve the content diversity in the network. However, this random cache decisions may affect the content popularity by caching the non-popular content; (b) *Popularity-based Caching* [13] that consists to cache the most frequently used and popular content instead of non-popular content, hence reduce the content access delay. However, deciding which content is popular is an open question especially in highly dynamic network; and (c) *Cooperative Caching* [14] that aims to enhance and improve content sharing, where the caching decision is applied under more than one administrative authority.

D. Quality of Service Support

Vehicular communication includes various types of content and traffic on top of dynamic and heterogeneity links. This communication requires a careful quality of service support to improve data delivery, optimize the resource utilization, and enhance network performance. Hence, some researchers are trying to enhance QoS support in VNDN. For instance, work in [15] proposes a data lifetime enhancement scheme to improve QoS, by adding a tag in data header, and classifying content into different types with different lifetime. This tagging and classification decrease the load on producer level and reduce memory construction cost. Works in [16], [17], and [18] design forwarding strategies that use in-network caching to improve the QoS and reduce the energy consumption. The idea consists of monitoring and estimating the bandwidth and other metrics and thus select the forwarding path to satisfy all demands. Moreover, work in [19] proposes a differentiated QoS mechanism on top of an existing congestion control shaping solution as a format of QoS. Four levels of priority have been defined, where each packet is assigned to one level. Even if these schemes account the QoS in the forwarding process, they didn't take in consideration the diversity and heterogeneity of VANET traffic, that plays an important role in the selection of optimal and efficient forwarding and caching scheme.

III. Network Model & Problem Formulation

A. Network Model

We model a multi-hop vehicular network by a graph $G = (N, L)$, where $N = \{C \cup P \cup I\}$ represents a set of nodes, and L a set of links between nodes. Each vehicle in the network can be a consumer C, a producer P, or intermediate nodes I that can cache content.

For a given cache store x, we denote W_x the maximum caching capacity. We split virtually the cache store x into κ sub-cache stores $x_i : x = \{x_1, x_2, \ldots, x_\kappa\}$. For each sub-cache store x_i, we denote φ_i the list of cached data, $k_i = |\varphi_i|$ the number of cached data, and w_i the maximum caching capacity.

For each cached data c, we denote s_c the content size, q_c the total number of issued demands, τ_c the required time before expiring the content from the cache, p_c the content popularity, and σ_c the content popularity-density value. Table I summaries some of the used notation across the paper.

TABLE I: Summary of notations

Notation	Meaning
N	List of nodes
L	Set of links
C	List of consumer nodes
I	List of intermediate nodes
P	List of producer nodes
W_x	Maximum caching capacity
κ	Number of sub-cache stores
x_i	i^{th} sub-cache store
w_i	Maximum caching capacity of x_i
φ_i	List of cached data in x_i
k_i	Number of cached content
s_c	Size of content c
q_c	Total number of issued demands
p_c	Popularity value
σ_c	Popularity-density value
τ_c	Time to cache
v_c	Profit of content c
α_c	Cache decision parameter

B. Problem Formulation

Due to the diversity of exchanged traffic in a vehicular network, deciding where to cache, which content to cache, which one to evict to keep room for other important content is a challenging task, especially when taking the cache store capacity and quality of service factors into consideration. Here, we formulate the caching problem as a Knapsack problem. The knapsack problem is a combinatorial optimization problem that aims to maximize (or minimize) some quantity while satisfying some constraints. For example, maximize the profit without passing the knapsack capacity, which is an NP-hard problem.

The objective function (1) aims to maximize the caching profit (v) for all contents to cache. α_c is a binary parameter to indicate if a content should be cached or not.

$$\max \sum_{c=1}^{k} \alpha_c v_c \tag{1}$$

Subject to:

$$\sum_{c=1}^{k_i} \alpha_c s_c \leq w_i, \quad \forall i \in \{1, \kappa\} \tag{2}$$

$$\sum_{i=1}^{\kappa} w_i \leq W_x \tag{3}$$

$$\alpha_c \in \{0, 1\} \tag{4}$$

Constraint (2) enforces that the size of all cached contents should not exceed the maximum caching capacity x_i for the sub-cache store c_i, while constraint (3) indicates the total of caching capacities for all sub-cache stores must be less or equal to the whole cache store caching capacity W_x. Finally, (4) is a non-negativity constraint that indicates the content selection:

$$\alpha_c = \begin{cases} 1, & \text{if the content } c \text{ is selected.} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

IV. QCR: A QoS-aware Cache Replacement Policy

In the following section, we introduce QCR, a QoS-aware Cache Replacement policy for vehicular networks. QCR aims to cache not only the popular content closer to consumers, but also the most important and long-term useful data that may not be expired in a short time. Consequently, improving the content retrieval, enhancing the overall user and network quality of service, and reducing the caching replacement operations. QCR, as illustrated in Figure 2, is based on two main steps: content classification and cache splitting, and content caching and replacing based on content popularity-density value.

A. Traffic Prioritization Scheme

Vehicular traffic can be categorized broadly into two types: traffic that is used only once or for a very short period of time (e.g., warning messages), and traffic that can be used multiple times (e.g., video streaming). Thus, the cache placement schemes and replacement policies must take content type into consideration.

Toward this, we propose a traffic prioritization scheme that aims to classify the traffic based on its type (cf. Figure 2). The type of content is a parameter that must be specified by the original content producer and added in the naming scheme [7]:

/Application-Type/Service-Type/Data-Name/Location/Time/Seq

Based on VANET scenarios and types of traffic, we define three *Application-Type* classes:

- Class A: This class covers safety applications such as emergency warnings messages, lane changing information, and collision avoidance information, etc. The content

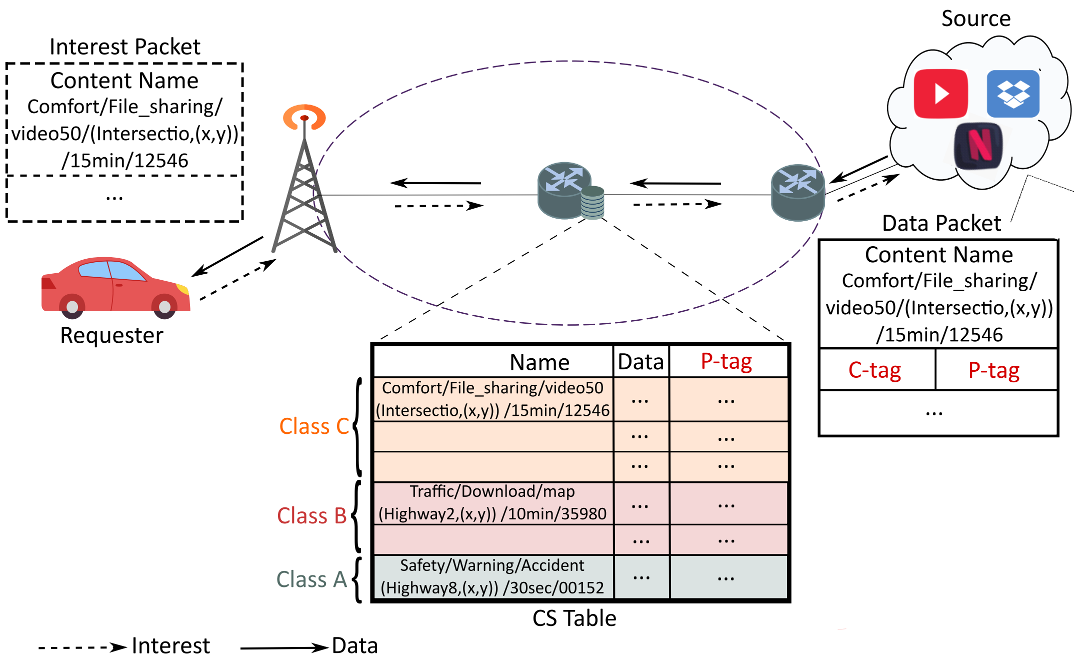

Interest Packet
Content Name
Comfort/File_sharing/video50/(Intersectio,(x,y))/15min/12546
...
Source
Requester
Data Packet
Content Name
Comfort/File_sharing/video50/(Intersectio,(x,y))/15min/12546
C-tag
P-tag
...
Name
Data
P-tag
Class C
Comfort/File_sharing/video50 (Intersectio,(x,y)) /15min/12546
Class B
Traffic/Download/map (Highway2,(x,y)) /10min/35980
Class A
Safety/Warning/Accident (Highway8,(x,y)) /30sec/00152
CS Table
Interest
Data

Fig. 2: QCR Working Mechanism.

in this type has a small time for freshness, and can be expired a short period of time.

- Class B: This class represents road traffic information that provides updated traffic information for drivers, road congestion information, and traffic map. The content of this class has a long lifetime before expired compared to the previous class.
- Class C: This class includes comfort applications traffics such as information about weather, gas stations, restaurants location, commercial advertisements, and interactive communications like Internet-on-the-move. The content of this class is characterized by larger lifetime before expired.

The content producer can specify the type of the content that will be included in the data packet under the field *C-tag*. This tag helps other nodes in the caching placement and replacement. Moreover, and as shown in Figure 2, we extend both data packet and CS table by adding *P-tag* to indicate how much the content is popular.

Cache Store Splitting: Based on the previous traffic classes, we divide the cache store into three sub-cache stores ($\kappa = 3$). However, the network administrator may define more classes according to their use case. Each data will be cached according to its *C-tag* to the appropriate sub-cache store. It is important to highlight that the caching capacity of each class may be different and defined according to the network conditions and memory resources. Here, we provide more space to Class C (comfort applications), then Class B (road traffic information), because its data has a long lifetime, and highly relevant for many vehicles. Moreover, Class A (safety applications) has a small caching space because it has a small size and a short time to expire. In addition, we add the *P-tag* in CS to indicate the content popularity-density value.

Data Packet Tagging: we extend the data packet with additional two fields: where the first one is *C-tag* contains the type of content class (i.e. A, B, or C), *P-tag*: represents the popularity-density of content which is the pillar factor to decide the caching replacement.

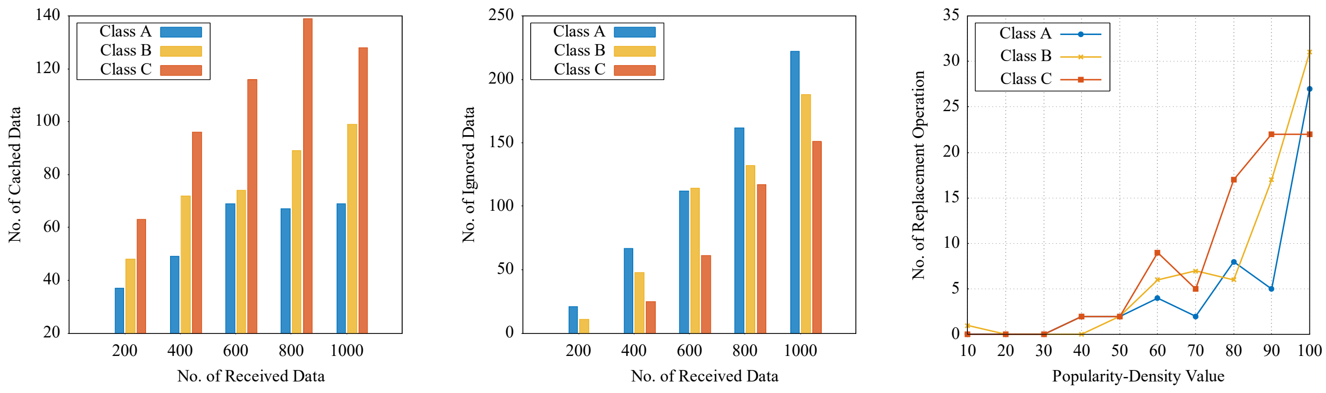
B. Content Replacement Policy

The proposed schemes are based on content popularity-density value, and have two phases:

Phase 1 - Calculate Content Popularity-Density Value: In the first phase, the content provider (or replica node) calculates the content popularity-density value, which is also included in the cache store and updated periodically based on the total number of issued demands for the content q_c and the remaining time to cache τ_c. The content popularity-density value is also sent in the data packet to inform other replica nodes about it. The content popularity is calculated as shown in Eq. 6, while the content popularity-density value is calculated as shown in Eq. 7 depending on content popularity and size. We calculate the content popularity-density in order to take the maximum number of contents that have high popularity value, whereby maximizing the cache profit. Some of the contents may have a big size and high popularity value, this may take all the space, which ends up with one content in the cache store. However, by taking both of size and the popularity of the content, the store may contain a lot of popular content.

$$p_c = \frac{q_c * \tau_c}{\sum_{i=1}^{k} q_i * \tau_i} \tag{6}$$

$$\sigma_c = \frac{p_c}{s_c} \tag{7}$$



(a) (b) (c)

Fig. 3: QCR Measurement Results.

q_c is the total number of issued demands for content c, τ_c the time to cache. k is the number of the contents in the cache store, and s_c is the size of the content c.

Step 2 - Caching Replacement Policy: In the second phase, the node may cache and/or replace the content based on its popularity-density value and the available caching capacity. The content with the highest popularity-density value is preferable to be cached to the appropriate sub-cache store. When a node receives a data packet, it checks first the *C-tag*. If there is a room in the associated sub-cache store, the content will be cached directly; otherwise, the algorithm extracts the received content's popularity-density value *P-tag*, then removes content with the lowest popularity-density value from the sub-cache store, and caches the received data without exceeding the allowed caching capacity. In case the received content's popularity-density value is the lowest value in the sub-cache store, the algorithm ignores the caching operation. Algorithm 1 represents the replacement policy.

V. Implementation & Evaluation

We have implemented QCR policy using Python programming language and evaluated it against existing NDN replacement policies including FIFO, LRU, and LFU in terms of the number of the cached data in the cache store, the number of ignored data due to the storage capacity, and the number of replacement operations. The overall scenario consists of defining three traffic classes (i.e. Class A, Class B, and Class C), as presented in subsection IV/A, hence splitting the cache store into three sub-cache stores. Class C has the largest storage capacity, followed by Class B, while Class A has the smallest storage. The distribution of demands follows the Zipf distribution. The evaluation has been performed on an Intel Core 5 Duo CPU at 2.4 GHz and DDR3 SDRAM of 8 GB.

Algorithm 1: QCR Replacement Policy.

```
Input: $d$: received content;
1 $\sigma_d$ := extract the content P-tag;
2 $i$ := extract the content C-tag;
3 for ($c \in \varphi_i$) do
4     if ($\sigma_c \leqslant \sigma_d$) and ($sum(w_i, s_d)) \leqslant w_i$) then
5         Replace $c$ with $d$;
6     end
7 end
```

Figure 3 shows QCR measurement results. In particular, Figure 3a represents the number of cached data in each class per number of received data. We can observe that the number of received data increases proportionally with the number of received data. Indeed, Class C has the largest number of cached data, this can be argued as its associated sub-cache store has the largest storage. Similarly, Class A has a small number of cached data because of the nature of content under such a class. Figure 3b shows the number of ignored data from the cache stores. Class A has a largest number of ignored data as most of traffic has sort lifetime and being cached is waste of storage. However, Class B and C respectively have less ignored data. This is argued by the fact that these data has a smaller popularity-density value compared to the cached one. Hence, caching them is pointless and may affect the overall quality of service. Finally, Figure 3c shows the number of cache replacement operation in the context of popularity-density value. As long as the popularity-density value increases the content must be cached. Hence, we notice that the behavior is growing exponentially.

Figure 4 shows comparison results against FIFO, LRU, and LFU. Figure 4a shows the number of cached data in each caching replacement policy, we can notice that FIFO, LRU, and LFU cache most of received contents (regardless of its popularity or class). However, QCR caches only the popular and most useful contents according to its class. The highest popular contents are cached where the non-popular and less frequently used contents are ignored. This result is proved in Figure 4b. When the popularity-density value increases, QCR is growing exponentially. However, FIFP, LRU, and LFU are

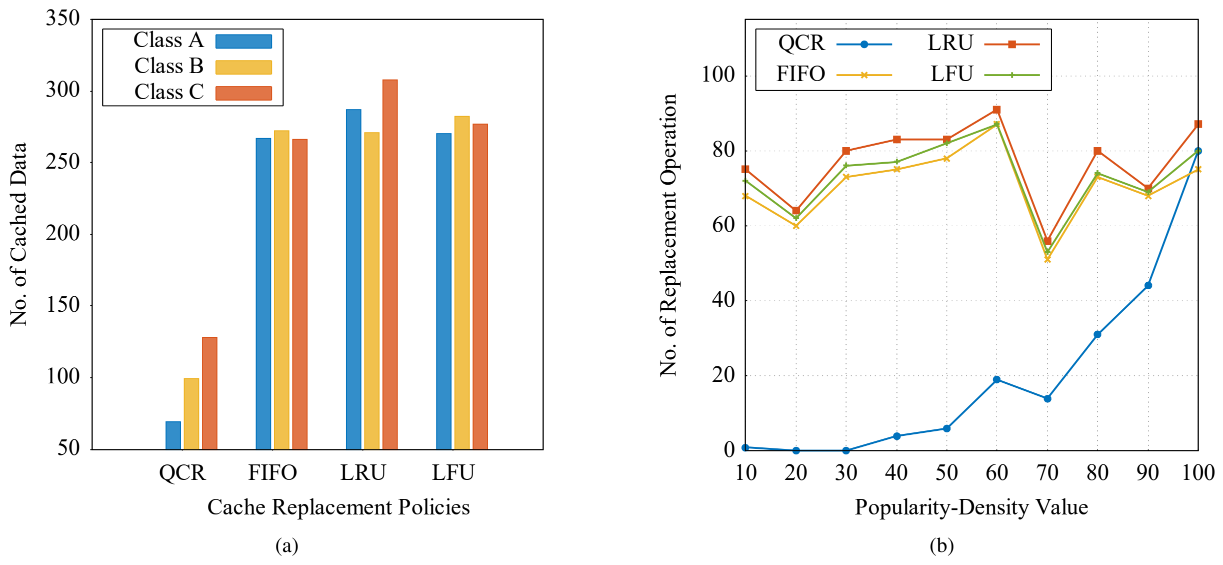


Fig. 4: QCR Comparison Results.

near to constant because they cache data without focusing on its popularity.

Finally, it is important to highlight that the caching decision is done during the content forwarding in an off-line manner. Which means that the content is forwarded downstream based on the PIT information (see Figure 1) and the caching operation is taking place. Hence, the time impact and complexity (depending on the number of cached contents) is not a critical issue.

VI. Conclusion

Traditional cache replacement policies may not work efficiently in VNDN networks due to the diversity of exchanged contents and traffic. Therefore, we proposed a QoS-aware Cache Replacement (QCR) policy. QCR assigns to each content a class, and splits the cache store into a set of sub-cache stores. Each sub-cache store handles one class of content. We modeled the problem as a Knapsack problem, and proposed a content popularity-diversity replacement policy that aims to select the content with the highest popularity-density to be cached and the lowest density to be evicted. A trade-off between content popularity and its size is taken into consideration to improve network quality of service. Extensive simulations prove the efficiency and scalability of the proposed solution.

Acknowledgements

The work of S. Luo was supported by the National 242 Project under Grant No. 2017A149. Dr. Luo is corresponding author.

References

[1] H. Khelifi *et al.*, "Named Data Networking in Vehicular Ad hoc Networks: State-of-the-Art and Challenges," *IEEE Communications Surveys and Tutorials*, 2019.

[2] B. Nour *et al.*, "A Survey of Internet of Things Communication using ICN: A Use Case Perspective," *Computer Communications*, 2019.

[3] H. Khelifi *et al.*, "A Name-to-Hash Encoding Scheme for Vehicular Named Data Networks," in *International Wireless Communications & Mobile Computing Conference (IWCMC)*, 2019.

[4] H. Khelifi *et al.*, "Security and Privacy Issues in Vehicular Named Data Networks: An Overview," *Mobile Information Systems*, 2018.

[5] B. Nour *et al.*, "Security and Privacy Challenges in Information Centric Wireless IoT Networks," *IEEE Security & Privacy*, 2019.

[6] I. U. Din *et al.*, "Caching in Information-Centric Networking: Strategies, Challenges, and Future Research Directions," *IEEE Communications Surveys & Tutorials*, 2018.

[7] H. Khelifi *et al.*, "An Optimized Proactive Caching Scheme based on Mobility Prediction for Vehicular Networks," in *IEEE GLOBECOM*, 2018.

[8] S. Ostrovskaya *et al.*, "Towards Multi-metric Cache Replacement Policies in Vehicular Named Data Networks," in *IEEE PIMRC*, 2018.

[9] A. Seetharam, "On caching and routing in information-centric networks," *IEEE Communications Magazine*, 2018.

[10] A. Ioannou *et al.*, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, 2016.

[11] G. Deng *et al.*, "Distributed Probabilistic Caching strategy in VANETs through Named Data Networking," in *IEEE INFOCOM WKSHPS*, 2016.

[12] G. Mauri *et al.*, "Optimal Content Prefetching in NDN Vehicle-to-Infrastructure Scenario," *IEEE Transactions on Vehicular Technology*, 2017.

[13] W. Zhao *et al.*, "An Efficient Cache Strategy in Information Centric Networking Vehicle-to-Vehicle Scenario," *IEEE Access*, 2017.

[14] L. C. Liu *et al.*, "CCN-based cooperative caching in VANET," in *IEEE ICCVE*, 2015.

[15] T.-Y. Wu *et al.*, "Data lifetime enhancement for improving QoS in NDN," *Procedia Computer Science*, 2014.

[16] Q. Huang *et al.*, "Ant-colony optimization based QoS routing in named data networking," *Journal of Computational Methods in Sciences and Engineering*, 2016.

[17] C. Li *et al.*, "Energy-efficient quality of service aware forwarding scheme for Content-Centric Networking," *Journal of Network and Computer Applications*, 2015.

[18] A. Kerrouche *et al.*, "QoS-FS: A new forwarding strategy with QoS for routing in Named Data Networking," in *IEEE ICC*, 2016.

[19] A. Alshahrani *et al.*, "A QoS Solution for NDN in the Presence of Congestion Control Mechanism," *International Journal of Advanced Computer Science and Applications*, 2016.