future internet

MDPI

Article

Quality-of-Service-Linked Privileged Content-Caching Mechanism for Named Data Networks

Shrisha H. S. [1,*] and Uma Boregowda [2]

[1] Department of Information Science and Engineering, Malnad College of Engineering, Visvesvaraya Technological University, Hassan 573202, India
[2] Department of Computer Science and Engineering, Malnad College of Engineering, Visvesvaraya Technological University, Hassan 573202, India; umaboregowda@gmail.com
* Correspondence: shree.1259@gmail.com

Abstract: The domain of information-centric networking (ICN) is expanding as more devices are becoming a part of connected technologies. New methods for serving content from a producer to a consumer are being explored, and Named Data Networking (NDN) is one of them. The NDN protocol routes the content from a producer to a consumer in a network using content names, instead of IP addresses. This facility, combined with content caching, efficiently serves content for very large networks consisting of a hybrid and ad hoc topology with both wired and wireless media. This paper addresses the issue of the quality-of-service (QoS) dimension for content delivery in NDN-based networks. The Internet Engineering Task Force (IETF) classifies QoS traffic as (prompt, reliable), prompt, reliable, and regular, and assigns corresponding priorities for managing the content. QoS-linked privileged content caching (QLPCC) proposes strategies for Pending Interest Table (PIT) and content store (CS) management in dedicated QoS nodes for handling priority content. QoS nodes are intermediately resourceful NDN nodes between content producers and consumers which specifically manage QoS traffic. The results of this study are compared with EQPR, PRR probability cache, and Least Frequently Used (LFU) and Least Fresh First (LFF) schemes, and QLPCC outperformed the latter-mentioned schemes in terms of QoS-node CS size vs. hit rate (6% to 47%), response time vs, QoS-node CS size (65% to 90%), and hop count vs. QoS-node CS size (60% to 84%) from the perspectives of priority traffic and overall traffic. QLPCC performed predictably when the NDN node count was increased from 500 to 1000, showing that the strategy is scalable.

Keywords: Named Data Network (NDN); information-centric networking (ICN); quality of service (QoS); Internet of Things (IOT)

Citation: H. S., S.; Boregowda, U. Quality-of-Service-Linked Privileged Content-Caching Mechanism for Named Data Networks. *Future Internet* **2022**, *14*, 157. https://doi.org/10.3390/fi14050157

Academic Editor: Abdelmadjid Bouabdallah

Received: 3 May 2022
Accepted: 18 May 2022
Published: 20 May 2022

1. Introduction

Internet technology has been transforming in the form of all-pervasive, ever-connected devices, which opens opportunities for applications and novel processes. Real-time and critical systems require high reliability and low response times in the application fields of Industry 4.0 [1], autopilot vehicles [2], disaster management, and emergency response situations. Networks often tend to use low power, lossy networks, and avail link-layer protocols such as LoRAWAN and NB-IoT [3], but quality of service (QoS) is not implicit, and the burden falls on the network layer to implement and achieve the desired reliability for data transmission.

The Named Data Network (NDN) [4] is based on an information-centric networking (ICN) paradigm, introduced as an additional protocol which complements internet protocol (IP). There are two important packets in NDN, namely, "interest" and "data". There are three components in NDN for stateful forwarding, namely, content stores (CS) [4], forwarding information bases (FIBs) [4], and Pending Interest Tables (PITs) [4]. The consumer node issues interest. The successive NDN node, upon receiving a consumer interest packet, performs a lookup in its CS for the requested content; PITs perform the same lookup for the content name entry. A CS is a memory of variable size used to store any data which flows

through the node temporarily. If the consumer request matches with the data available in the CS, then the interested data is served; otherwise, the content name, along with its associated incoming interface (information of neighboring node from which the interest packet was received), is recorded in the PIT. If the content name entry already exists in the PIT, indicating that there are other consumers waiting for the said content, then the incoming interface of the new interest is added to the content name entry of the PIT. The unsatisfied interest packet is forwarded to next node. An FIB maintains the interfaces of the neighboring nodes. The content names in the PIT are alive until the request is either satisfied or dropped due to expiry of time, pre-emption, etc. The sequence of interest-packet forwarding repeats until the interest reaches either the source of the content, or any CS of intermediate nodes where valid content is present. Once the content is found in an NDN node, the data packet (with requested content) traverses the reverse path by referring to the PIT entry and its associated interfaces to reach the consumers. The CS of the NDN nodes along the path may decide to cache the traversing content, depending upon their admission control strategy. The unique feature of an NDN is that every time a consumer requests content, the request need not be satisfied by the producer of that data. Intermediate nodes may have cached the data in their CS, and the data can be served directly from a CS. Content names are organized in a hierarchical manner, and are addressed as depicted in Figure 1 [4].

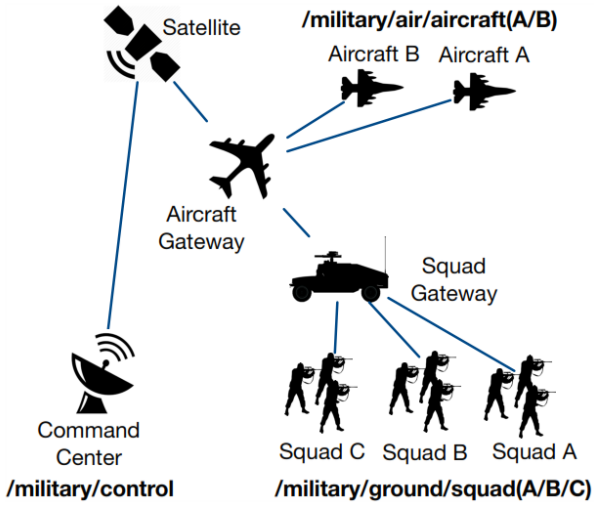


Figure 1. NDN operational scenario [4].

There is increasing research towards employing NDNs for communication network applications [5–7]. Research studies [8,9] conclude that NDNs can constitute reliable network-layer protocols for networks with acceptable network throughput and latency. A survey of the existing literature shows that there is very little exploration on the QoS properties of NDNs from the context of content caching for networks which use a low-power network architecture, such as the Internet of Things (IOT). In an IP network, QoS involves managing network resources such as bandwidth, buffer allocation, and transmission priority [10]. An NDN-based network adds an additional dimension of resource management, i.e., CS management and PIT management.

The implementation of QoS for a network involves assuring resource priority to the qualified content stream over the regular content stream. Flow-based QoS guarantees the priority for a content stream based on the application of origin, and class-based QoS guarantees the priority for a content stream based on traffic classes such as telephony, network control packets, etc. [11]. In NDNs, content may have originated from multiple sources, and may be consumed by multiple destinations. For this reason, content-forwarding resources are distributed, and these NDN properties make QoS enforcement difficult. The request–response sequence in an NDN navigates in a hop-to-hop manner, which may trigger an unforeseen amount of data packets and quickly exhaust the resources. Pre-

empting pending interests, dropping data packets, and prioritizing the interests of QoS by differentiating regular traffic without a proper strategy may lead to a burst in interest packet volume, and resources may become underutilized [12].

For NDNs, QoS strategies can leverage the inherent advantages of a CS. A CS improves turn-around time, reduces traffic congestion by availing the content in demand through multiple NDN nodes, and acts as a re-transmission buffer. QoS strategies should consider managing PITs, which indicate pending interests to be satisfied. As priority interests become satisfied by data packets, regular interests recorded in a PIT may be starved. In another situation, a CS and PIT may become saturated with QoS traffic and become unable to cache further content and maintain the record of pending QoS content interests, respectively. In such situations, heuristic mechanisms are needed to manage the CS and the PIT with proper service balance for the regular and competing QoS content. While managing distributed resources such as PITs and CS, coordination between NDN nodes are essential. Identical content in a cluster of NDN nodes decreases the potential of the network [13]. The overall CS capacity of the NDN may be efficiently used if a cooperative caching scheme is implemented. In the context of PITs, a saturated PIT may pre-emptively remove entries and terminate the path for the content flow. This wastes the content-forwarding resources of preceding NDN nodes. This is unacceptable in cases where the traffic is QoS guaranteed. In this light, this proposal explores ways to manage the PITs and CS of NDNs to implement QoS in NDNs.

The Internet Engineering Task Force (IETF) has published guidelines on tackling QoS issues in information-centric networks using the techniques of flow classification, PIT management, and content store management from a resource-management point of view [14–16]. The traffic flows are identified as, and in the order of $(prompt, reliable) > prompt > reliable > regular$, for "prompt" QoS, which aims to minimize the delay for QoS traffic by providing a content-forwarding priority. For "reliable" services, the priority order is $(prompt, reliable) > reliable > prompt > regular$, with the intention of guaranteeing the reliable delivery of data packets to consumers. The nomenclature "prompt", "reliable", and their priority orders are preserved in QLPCC, as proposed by the IETF publication. The flow classification introduces the concept of embedding name components to establish equivalence among different traffic flow priorities [14–16]. The proposed QoS-linked privileged content-caching (QLPCC) mechanism employs this concept to facilitate the assignment of priorities for different traffic flows indicated by the Flow ID. A Flow ID is a name component which uniquely identifies a priority traffic flow for the implementation of flow differentiation. QLPCC proposes the calculation of an eviction score for a given priority content entry, for eviction from a saturated PIT. Content store management consists of an admission-control phase, where heuristics decide whether or not to cache the content. Each content belongs to a traffic flow, and each QoS node adopts a Flow ID. If the content belongs to the adopted Flow ID, then it is cached by the CS; otherwise, the content is forwarded without caching. The admission-control algorithm defines the steps to adopt new Flow IDs and to drop the old ones. The second phase of CS management is content eviction for the better utilization of limited available memory. QLPCC proposes a novel heuristic-based content-eviction algorithm by considering their QoS priorities, usage frequency, and content freshness. Time-expired content is evicted first, followed by prompt-priority content which was overlapped by Least Frequently Used (LFU) and Least Fresh First (LFF) algorithms. If such an overlapping was not found, then LFU prompt-priority content is evicted. If a memory requirement still persists, then the content-eviction strategy applies LFU and LFF algorithms on reliable priority content. Content that is overlapped by LFU and LFF is evicted first, followed by LFU reliable-priority content.

The proposed QLPCC designates dedicated NDN nodes as QoS nodes for implementing the proposed strategies. QoS nodes are resourceful NDN nodes that handle QoS traffic flows as per the QLPCC strategy. A QoS node can be any resourceful NDN node, including routers. The unique feature of the proposed QLPCC is that it provides support for privileged content through both PIT management and CS management. The distinguishing features of QLPCC, when compared with other schemes, are provided in Table 1.

Table 1. Uniqueness of QLPCC when compared with existing schemes.

Schemes	QoS Support	PIT Management	CS Management
LCE [17]	No	No	Yes
SDC [18]	No	No	Yes
SNN [19]	No	No	Yes
Probability cache [20]	No	No	Yes
GrIMS [21]	No	No	Yes
FairCache [22]	No	No	Yes
CPHR [23]	No	No	Yes
HLLR [24]	No	Yes	No
IFS [25]	No	Yes	No
PRP-MC [26]	No	Yes	No
PIT lifetime mechanism [27]	No	Yes	No
EQPR [28]	Yes	Yes	No
PRWR [29]	Yes	Yes	No
PRR [29]	Yes	Yes	No
QLPCC	Yes	Yes	Yes

QLPCC is simulated on an ndnSIM [30] platform, and results are compared with EQPR [28], PRR [29], probability cache, and LFU and LFF schemes. QLPCC outperformed the previously mentioned schemes in terms of content store hit rate, response time, and hop count reduction from the perspective of priority traffic and overall traffic. QLPCC is also evaluated in terms of content store hit rate vs. the percentage of QoS nodes in the network.

The contributions of the proposed QLPCC strategy are listed as follows:

- The QLPCC strategy to manage QoS content in NDN-based networks;
- A QoS-based PIT management scheme through a novel Flow Table involving the calculation of eviction scores;
- A QoS-based content store management scheme, which proposes the content store admission-control and content-eviction heuristics;
- A priority flow adoption method for caching QoS-linked privileged content among QoS nodes;
- A reduction in hop count, an increase in content store hit rate, and an improvement in response time.

The paper is organized as follows. Section 1, above, provided an introduction to the work, and will be followed by a brief discussion of the QoS guidelines published by the IETF, with respect to ICN, in Section 2. Section 3 overviews the related works in terms of NDN traffic classification, content store management policies, and PIT management policies. Section 4 provides a description of the QLPCC strategy in terms of Flow Table description, the QoS table, eviction score calculation, PIT management, content store management, and finally, an illustration through a case example. Results and discussions are provided in Section 5, followed by a conclusion in Section 6.

2. NDN and QoS

Research groups under the IETF have explored the issues of traffic flow classification [14], congestion control QoS services for ICNs [15], and resource management for content with different priorities [16]. Sections 2.1–2.3 provide the takeaways for the same issues.

2.1. Flow Classification

Flow classification is the foundation of QoS. Flow classification is a problem of grouping the packets and identifying the priority at which forwarding resources must be allocated for the said group [14].

- In an IP network, flow classification is executed using a source address, destination address, source port, destination port, and protocol type identified with the packet.

NDN packets cannot be identified by their source and destination addresses for flow-identification purposes. Therefore, an alternative mechanism has been proposed by the Internet Engineering Task Force (IETF);

- The equivalence class name component type (ECNCT) [15] introduces name components which identify a particular flow and infer the equivalence classes of the traffic. This mechanism does not need any alterations in the NDN paradigm and easily integrates with the existing framework. The name component can be encoded at any granularity of the name hierarchy of the content, and this facility can be used to identify equivalence classes of streams and sub-streams of content at any desired granularity.

Consider the named content "Netflix/Show1-Prompt/Frame ID/#Segment", represented in Table 2: all content of the streaming service Netflix may not require QoS, but a subset of the content may. An equivalence class identifier with encoded naming conventions can establish the equivalence class without any additional overhead if the naming conventions are recognized by the network stakeholders. The content consumer issues "interest packets" to the NDN network, indicating the equivalence class. The content producer/intermediate NDN node streams "data packets" indicating the corresponding equivalence class which may be served to multiple destinations, either through a direct path or through a CS in a non-synchronized way. The ECNCT does not add any additional overhead to the existing packet structure, but requires the addition of a "Flow Table", which can identify the name prefixes for establishing class equivalences. Flow classification has useful applications for enforcing forwarding-rate control for the content of equivalent classes, the estimation of unique flows traversing through a given bottleneck, which is useful in congestion control, and to make caching decisions.

Table 2. An example of an ECNCT mechanism.

Netflix	Show1-Prompt	Frame ID	#Segment
Regular name	ECNCT-encoded name	Regular name	Regular name

2.2. Issues to Consider for Implementing QoS Services for NDN

- Congestion control in an NDN is aimed at preventing network overload, preventing the starvation of a particular class of traffic, and implementing fairness in resource allocation;
- Leveraging the NDN name hierarchy is beneficial for traffic classification, rather than framing a separate definition. The network may use CS as an instrument for implementing temporary re-transmission buffers and avoiding content request load on the producers;
- The resources which can be managed to implement QoS in an NDN are bandwidth, content stores, and PITs [15];
- A content name entry into the PIT ensures sufficient bandwidth is allocated to the content in the inverse path towards the consumer, but this time-invariable interest entry may have to wait for a long time to become satisfied by the corresponding data packet, causing inefficient PIT space and bandwidth reservation;
- For managing NDN resources, policies for the identification of traffic equivalence classes and their corresponding treatment must be specified;
- As the consumer-requested content may be satisfied by multiple sources, the effect of topology plays an insignificant role for QoS;
- QoS mechanisms of IP cannot be directly ported to NDN because hop-to-hop transmission is not confined to a single path for an NDN, thereby restricting the ability of advanced resource allocation during the process of network admission control.

2.3. Resource Management for Prioritized Content

(Prompt, reliable), prompt, reliable, and regular are the four flow priorities recognized for the implementation in QoS [16]. Two forwarding queues, one for prompt forwarding and another for regular forwarding, are utilized. In this light, three situations of decision making in the context of content forwarding and caching are realized:

- Local cocooned decisions, which are not inter-related with any other resources;
- Decisions based on locally related network resources;
- Decisions based on globally distributed resources.

Local cocooned decisions do not refer to the status of other resources or mechanisms while making caching or forwarding decisions. Content is allotted to the respective queues according to their priorities. Reliable-priority content is provided prominence over regular-priority content for caching purposes. In PIT management, prompt-priority entries will replace regular-priority entries if the PIT of the NDN node is saturated. In situations where decisions are based on locally related network resources, caching and PIT management operations are based on the validity of a PIT entry and the status of the prompt forwarding queue of a given NDN node. Here, the word "local" refers to intra-device resources. If the arriving content complements a PIT entry, the said content is forwarded with reference to its priority. In case the prompt queue is full, the prompt-priority content will be assigned to regular queue and provided priority over regular content. Caching decisions follow the order of (prompt, reliable), reliable, prompt, and regular, respectively, while recognizing priority levels for adjusting the weights for content-caching algorithms. From the view point of globally distributed resources, the focus is on maintaining uniformity across PIT and CS management schemes in terms of QoS policies.

3. Literature Survey

3.1. NDN Traffic Equivalence Class Classification

NDNs host regular traffic and priority traffic with diverse resource and time requirements. This statement is true for different deployments of NDNs, such as IOT, wireless sensor networks (WSNs), mobile ad hoc networks (MANETs) [31], etc., where packets originate from endpoints such as sensors, actuators, or from cloud-hosted applications through network gateways. It is necessary for NDNs to classify traffic based on equivalence classes when different traffic classes demand differential treatments in terms of resource allocation (content store memory) and forwarding preferences (PIT content name entry). Due to the distributed nature of content services, an IP-like traffic classification based on addresses and ports become unfeasible, since NDNs abandon the IP address-centric paradigm and proceeds towards a multi-source content service backed with NDN content stores [32].

A unique characteristic of NDN content is that it is addressed through hierarchically structured and unique names for a given namespace [33]. NDN content names find a place in both interest packets and data packets. The property of all pervasive hierarchically structured NDN content names provides an opportunity to identify and classify the NDN traffic flows. Therefore, this paper [14] proposes NDN name-based flow classification using an ECNCT mechanism, as discussed in Section 2.1. The mechanism integrates with an NDN framework without any major modifications, and the strategy is computationally simple.

This proposal [14] considers two QoS dimensions, i.e., delay and reliability. The service name dealing with delay is called "prompt", and the service name dealing with reliability is called "reliable". QoS priorities are assigned to traffic flows based on the presence of ECNCT in the content name hierarchy, corresponding to the traffic flow granularity at which QoS has to be enforced. Service classes are given priority, as shown in Figure 2. Traffic flows with both prompt and reliable service classes receive the highest priority, followed by the prompt, reliable, and regular service classes, respectively. For example, with $Netflix/Show1 - Prompt/FrameID/\#Segment$, assuming a particular content has QoS requirements, the $Show1 - Prompt$ name will indicate the equivalence class identified

by the Flow Table installed in the NDN nodes, and all the data packets corresponding to that $Show1 - Prompt$ will receive a defined QoS treatment. Flow Tables are updated across the network on every update cycle.

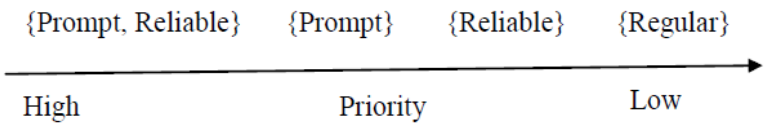


Figure 2. Priority scale for different service classes.

3.2. Content Store Management Policies

In an NDN, the CS acts as temporary memory installed in the NDN nodes for the storage of the data packets routed through them. Content-caching algorithms are application-independent [34], and should answer the questions of "whether to cache the content" and "which content should be evicted for freeing up memory". Content and its caching location determines the efficiency of the NDN network in terms of delay, resource usage, power consumption, congestion, etc. Heuristic content-caching algorithms outperform non-heuristic approaches, and a wealth of literature provides different families of algorithms which consider different parameters, such as content name, expiry time, producer, consumer, time to live value, etc. Some algorithms consider network topology as one of the criteria for content caching, but these are computationally expensive and do not reflect real-time changes in the network topology [35]. Another category of heuristic algorithms compute caching decisions based on the content present in the neighboring CS of an NDN node cluster [36], and some approaches take isolated caching decisions using locally defined rules [37]. The aim of CS management schemes is to increase the CS hit ratio and decrease the hop count between the consumer and the content [38]. The caching algorithms may follow a non-cooperative model or a cooperative model.

The non-cooperative model is characterized by the idea that caching-decision heuristics at the CS are isolated and independent, without considering the status of other NDN nodes such as a Leave Copy Everywhere (LCE) [17] cache or First-In-First-Out (FIFO) queue schemes. In, SDC [18] CS of NDN nodes are assigned binary numeric ID. Content names whose hash values coincide with the binary numeric ID of the NDN nodes cache the corresponding content. Stimulable Neural Networks (SNNs) [19] predict the relationship between interest packets and cache the corresponding interrelated data packet. The algorithm does not take the interest packet relationship between multiple nodes and only considers the current NDN node. SNNs have a trade-off between real-time prediction accuracy and computational overhead, which may contribute to latency. The probability cache [20] scheme is straightforward and adopted widely, and indicates the probability with which a given content will be requested by the consumer for a future time frame. A static probability value P_C, based on the frequency of a request for a given content C, can be defined as the ratio of the frequency of a request for a given content to the total number of content requests made to the content store, as shown in Equation (1).

$$P_C = \frac{Frequency_{request}(C)}{Frequency_{request}(ContentStore)} \quad (1)$$

A variant of probability caching calculates the global popularity based on the frequency of a request and the location of consumers across the network [39].

In the cooperative model, the CS employs heuristics which consider the status of neighboring NDN-cached content to decide whether or not to cache a data packet. GrIMS [21] identifies the neighboring NDN nodes of the consumer who requests the content. If the path from the content supplier/producer to the consumer captures any of the identified neighbors, then that NDN node will cache the content. FairCache [22] aims to develop a caching mechanism which ensures fairness in service delay to all content consumers

through a Nash bargain, where each CS tries to increase its "utility" by striking a balance between caching the content all by itself and redirecting the content interest to the next NDN node. CPHR [23] assigns memory chunks between different contents based on their hash functions, with the aim of inducing semi-autonomous behavior from the collaborating NDN nodes. The heuristic function executes this operation as a min-max optimization and continuously evolves to maximize the CS hit rate by distributing the data efficiently across the network through a hash routing method. NICE [40] calculates a threshold value, called "betweeness", taking into account the presence of the content in multiple paths, its popularity, and the location of caching relative to the consumer. The threshold value determines the action of caching or dropping the given content and reduces the possibility of multiple redundant presences across paths.

3.3. PIT Management

A PIT dictates the traffic flows in an NDN by providing a path from a producer to a consumer who has requested the content, and is the cornerstone of stateful forwarding. The magnitude of a PIT determines the maximum number of NDN traffic flows that can be managed. If there is no coherence between the PIT of NDN nodes along the path between the producer and consumer, then the traffic path may terminate and result in a wastage of the forwarding resources of predecessor NDN nodes. A PIT records an entry when an interest packet for the content is received by the NDN node, but it is neither the producer of the content nor can its CS satisfy the request; therefore, it forwards the interest packet to the next NDN node. The entry remains in a PIT until either the content request is satisfied by the NDN node, by receiving the requested content routed through its path, or the entry is pre-emptively removed due to a predefined policy, such as time expiry, resource allocation to high-priority traffic, etc.

PIT saturation occurs when more interest entries are required to be recorded than what is allowed by the PIT's size. In the scenarios where a PIT reaches its maximum capacity, incoming interests which cannot be satisfied by the local CS may be dropped. This remedies the event of active traffic flow path termination for existing entries. Ignoring the interest packet entries due to PIT saturation may increase the transmission delay, if that NDN node is located along the shortest path between the content producer/supplier and the consumer. To mitigate this delay and re-transmission efforts for QoS traffic, strategies need to be developed for maintaining the coherence between the PITs of NDN nodes along the path of the QoS traffic.

There is a need for a PIT entry replacement policy when full and new interests arrive at NDN nodes. HLLR [24] identifies an entry which has a long lifetime and satisfies the least number of content request interfaces and replaces it with the new one. The algorithm calculates a value called "price", which is the ratio of the PIT entry's lifetime to the number of request interfaces. The entry whose price is higher is evicted. An intelligent forwarding strategy [25] identifies PIT entries which contribute to network congestion based on the number of interests that are satisfied for a particular entry. When the number of interest entries increases in a PIT, the entry which has been pending for a long time and that is not satisfied by returning data is judged to be a congestion contributor and evicted. PRP-MC [26] does not remove a satisfied entry from the PIT, as in the case of a standard NDN. Even after satisfaction, the interest entry remains active, securing the path for content with same prefix names. The proposal takes a proactive approach of calculating a "contribution" value, which is an aggregation of the number of content interests satisfied by a group of interest entries with the same prefix names. This approach mimics the popularity-driven methods for interest entry eviction in PITs. A PIT lifetime mechanism [27] recognizes the problem of the PIT memory exhaustion phenomenon, even after measures such as interest entry eviction, after assigning a small time to a live value, and proposes a bounded hop-count method. The interest packet with more than the threshold distance indicated by the hop count is dropped, preferring, therefore, the interests from the nearest locations. This strategy demonstrates the reduction in packet-drop events.

EQPR [28] manages the QoS traffic through PIT entry strategies based on packet priorities and estimated turn-around times. If the PIT is full, then non-priority entries are removed. Moreover, if the estimated turn-around time of a packet is greater than remaining lifetime of the PIT entry, then the given PIT entry is evicted. The PRWR [29] allows PIT entries, regardless of their QoS priority, when the PIT memory space is not full. Once the PIT is saturated, priority-interest packet entries replace non-priority-interest packet entries. PRR [29] reserves a dynamically adjustable portion of the PIT for priority-interest packets exclusively.

EPQR [28], PRWR [29], and PRR [29] strategies do not consider the distinction among priority data streams, i.e., prompt, reliable, and regular traffic; therefore, they lack the ability for arbitration among inter-priority packets. The previously mentioned strategies do not possess the ability to federate among the NDN nodes in terms of caching the priority content. QLPCC addresses these two disadvantages in handling QoS content in NDN networks.

4. QoS-Linked Privileged Content Caching

The QLPCC manages the QoS traffic through dedicated QoS nodes. QoS nodes are embedded as intermediate NDN nodes between a consumer and a producer, specifically for handling QoS contents in terms of content caching and PIT management, as proposed in the QLPCC strategy. Other non-QoS NDN nodes do not differentiate between regular and privileged traffic, and handle them according to standard PIT and CS policies. This method benefits regular traffic, allowing it to obtain its fair share of network bandwidth. This proposal introduces the implementation of Flow Tables for PIT management and QoS Tables for content store management policies of content eviction and content-admission control, as novel proposals to be implemented in QoS nodes. The Flow Table identifies the content name, extracts the Flow ID, determines the equivalence class, and calculates the eviction score. For each traffic flow, a Flow ID is assigned to identify the content streams and to provide a continuous streaming path through the QoS node. The eviction score provides a measure of importance that the PIT entry has among the content of the same QoS privilege. This helps PIT eviction strategies to decide when there are overlapping candidates with the same degree of privilege. The PIT entry-eviction strategy is presented in Algorithm 1. A detailed description of the working of the Flow Table, along with the PIT entry-eviction algorithm, is provided in Section 4.1.

Algorithm 1 PIT entry-eviction algorithm.

```
Evict all TTL-expired interest entries
for Regular content names in PIT do
    Evict the content name entry with the highest eviction score
      return evicted content name
end for
for Reliable content names in PIT do
    Evict the content name entry with the highest eviction score
      return evicted content name
end for
for Prompt content names in PIT do
    Evict the content name entry with the highest eviction score
      return evicted content name
end for
```

The QoS Table identifies the content name, extracts the Flow ID, monitors the content time stamp and expiry time, and documents the path of the QoS nodes where a given Flow ID is adopted. Content names are used to extract Flow ID and QoS privilege. The content-eviction algorithm examines the expiry time for evicting the obsolete content and freeing up the memory. QoS privilege provides the order in which the content has to be considered for eviction. The content-eviction strategy, presented in Algorithm 2, is called when the

CS memory is exhausted and there is a need to free up the limited available memory. The content-admission-control strategy is presented in Algorithm 3. A detailed description of the working of the QoS Table, along with the content-eviction and admission-control algorithms, is provided in Section 4.2.

Algorithm 2 Content-eviction algorithm.

```
Evict all time-expired obsolete content
for Prompt content in content store do
    Apply LFU
    Apply LFF
    Evict the content overlapped by LFU and LFF, followed by LFU content
     return evicted content name
end for
for Reliable content in content store do
    Apply LFU
    Apply LFF
    Evict the content overlapped by LFU and LFF, followed by LFU content
     return evicted content name
end for
```

Algorithm 3 Content-admission-control algorithm.

```
if Content Store memory full then
    Call content-eviction procedure
end if
if Flow ID adopted by content store then
    Cache the content
else
    Issue interest packet for QoS table update
    Lookup the QoS table for the QoS node which has adopted the given Flow ID
    if FlowID adopted ? then
        Forward the content
    end if
    if QoS table==full then
        Forward the content to be adopted in future QoS nodes
    else if Flow ID has not been adopted then
        Adopt the Flow ID
        Set Flow ID expiry time to the longest TTL value among the content belonging to
Flow ID
     return Flow ID
    end if
end if
```

4.1. Flow Table and PIT Management

The purpose of the Flow Table is to maintain a record of equivalence classes of the PIT entries that are currently active through the given QoS node. Table 3 provides the format of the Flow Table attributes. The Flow ID and equivalence class ID are extracted from the content name hierarchy, where the granularity of QoS is implemented. Eviction score is a function that generates a value that assists with evaluating the candidate content for PIT eviction in cases of PIT saturation. The novelty of eviction-score assignment is the ability to arbitrate the priority between the same level of QoS privilege. Though the IETF documents [14–16] abstractly order the privileges into (prompt, reliable), prompt, reliable, and regular services, respectively, eviction score acts as an additional tool to handle inter-prompt, inter-reliable PIT entries which are not addressed by the said IETF publication.

Table 3. Flow Table format.

Content Name	FlowID	Equivalence Class ID	Eviction Score
Netflix/Show1-Prompt/Content1/S1	Show1-Prompt	Prompt	31.59
Temperature/Sensor1-Reliable/A/S11	Sensor1-Reliable	Reliable	19.28
Messenger/User1-Reliable/S2	User1-Reliable	Reliable	21.29
Netflix/Show1-Prompt/Content1/S1	Show1-Prompt	Prompt	44.76

The ECNCT [15] name component for QoS is embedded in the name hierarchy at the granularity level of QoS enforcement and identified using the naming convention "name—Equivalence Class ID". The application searches for the name—Equivalence Class ID string pattern in the content name and extracts the Flow ID and equivalence class ID to populate the Flow Table. Each content name is associated with the eviction score value, as per Equation (2).

$$EvictionScore = \frac{TTL}{r_c} \quad (2)$$

where TTL represents the time To live for the given PIT entry, and r_c represents the number of request interfaces (indicates the number of consumers waiting to be satisfied) associated with the given PIT entry. A higher eviction score represents a higher probability of PIT eviction for a given equivalence class. The PIT eviction strategy selects an entry with a higher eviction score to evict among the same equivalence class, when there is a requirement for more memory after the time-expired entries are removed from the PIT. The eviction score mechanism acts as an arbitrator between contesting QoS-privileged interests in a resource-constrained PIT. The steps to evict PIT entries by referring to the Flow Table and using the eviction score are provided in Algorithm 1.

Algorithm 1 first searches for TTL-expired PIT entries and evicts them, followed by the eviction of PIT entries of regular content in descending order of eviction score. Further memory requirements are satisfied by evicting PIT interest entries of reliable-equivalence-class in descending order of eviction score, followed by prompt-equivalence-class PIT entries in descending order of eviction score. The content names of the (prompt, reliable) equivalence class, indicating both prompt and reliable privileges, are evicted only on TTL expiry.

4.2. QoS Table and Content Store Management

Content store management deals with admission-control techniques for content caching and cached-content-replacement strategies. Admission control is the process of determining whether or not to cache the content. Replacement strategies dictate which content has to be replaced in the scenario of CS memory exhaustion. Admission control for the content follows the priority order $(prompt, reliable) > reliable > prompt > regular$ [14,16]. CS management follows a federated approach for caching the content, with the aim of eliminating redundancy and saving precious memory space. QoS nodes implement the QoS Table, which is used to arbitrate the decisions of admission control and content replacement. Each CS of a given QoS node federates itself with other QoS nodes, based on traffic flows which are distinguished based on Flow ID. If a given flow is already being adopted by a QoS node, then content belonging to that Flow ID is cached by that QoS node. The format of the QoS Table is shown in Table 4.

Table 4. QoS Table of q_3.

Content Name	Flow ID	Expire Time	Time Stamp	QoS Node Path of Adopted Flow ID
Netflix/Show1-Prompt/Content1/S1	Show1-Prompt	8:15 PM	7:59 AM	q_3
Temperature/Sensor1-Reliable/A/S1	Sensor1-Reliable	1:15 PM	12:35 PM	n_8, q_2
Messenger/User1-Reliable/S11	User1-Reliable	3:05 PM	2:40 PM	n_8, q_2, n_2, n_1, n_{10}, q_4
Netflix/Show1-Prompt/Content1/S1	Show1-Prompt	2:55 PM	2:45 PM	q_3

CS memory is limited and may become exhausted with time. Therefore, to free up the memory, a content-eviction strategy is proposed in Algorithm 2. The content-eviction algorithm considers content freshness, usage frequency, and content privilege as criteria to evaluate the suitability for eviction. Expired content is evicted in the first pass, irrespective of its QoS privilege. Upon further requirements for memory, the strategy applies LFU and LFF on "prompt" content. Content that is overlapped by both algorithms is evicted first, followed by LFU content. If a further memory requirement persists, then "reliable" content is evicted based on overlapping LFU and LFF algorithms, followed by LFU content evictions. The content with (prompt, reliable) status is evicted only upon expiry.

An admission-control strategy is proposed in Algorithm 3. Admission-control algorithms evaluate the suitability of content to be cached in the CS of a QoS node. If the CS memory is exhausted, the algorithm calls the content-eviction procedure. The content streamed through the NDN are part of a traffic identified by a Flow ID. The Flow ID is extracted using content name, as illustrated by the QoS Table in Table 4. Each QoS node caches the content in its CS, depending upon whether a given content belongs to an adopted Flow ID. If the Flow ID is adopted by the QoS node, then the content is cached. If the Flow ID is not adopted by QoS node, then an interest packet is issued to te NDN for the QoS Table update, which will indicate the latest status of the Flow ID adoption. The QoS table is searched to identify the QoS node where the Flow ID is adopted, and the given content is forwarded. If the Flow ID is not adopted by any node, and if the QoS table is not yet full, the current QoS node adopts the Flow ID and sets an expiry time for the adoption, which reflects the longest TTL value of the content belonging to the Flow ID. If the QoS node cannot adopt the Flow ID due to memory requirements, it forwards the traffic towards further QoS nodes for adoption.

4.3. An Illustration

A computer network comprising IoT sensors, actuators, and resourceful NDN nodes (routers/gateways) interconnected by a wired or wireless medium is considered as an experimental scenario. All nodes of the experimental network are presented as $M = \{m_1, , m_2, m_3, \ldots m_k\}$, where the content producers $P = \{P_1, P_2\}$, content consumers $C = \{C_1, C_2, C_3, C_4\}$, and QoS nodes $Q = \{q_1, q_2, q_3, q_4\}$ all belong to M, i.e., $P \in M$, $C \in M$, and $Q \in M$. The proposal caches the QoS content from the producers in QoS nodes. The producers generate both QoS and non-QoS data. The QoS data are cached by QoS nodes, and non-QoS data are cached in regular intermediate nodes (no privileged treatment). Let the QoS contents produced by producer nodes P be denoted as $QContent = \{qc_1, qc_2, qc_3, \ldots qc_i\}$, and let $|QContent|$ show the amount of QoS content cached in the content store of a given QoS node. Content consumers issue and propagate the interest packets towards content producers. The set of interest packets that reach the NDN node at time t is denoted by $Interest(t) = \{I_1(t), I_2(t), \ldots I_j(t)\}$, and $ContentStore(m_k, t) = \{(qc_1, t), (qc_2, t), \ldots (qc_i, t)\}$ represents the content store content at time t for an NDN node m_k, $Served(Interest(t), ContentStore(m_k, t))$ represents the number of interests packets that are served for t, and the cached content hit rate is the ratio of the

number of interest packet served by the content store to the total number of interest packets received and calculated using Equation (3).

$$HitRate(ContentStore) = \frac{|Served(Interest(t), ContentStore(m_k, t))|}{|Interest(t)|} \quad (3)$$

An illustration of a case example of the proposed QoS-linked collaborative content caching is presented in this section, using the network scenario represented in Figure 3. Suppose that QoS content produced by P_1 $Netflix/Show1 - Prompt/Content1/Segment1$ is requested by consumer C_1 by issuing an interest packet and forwarding it to node q_2 through n_7. The content is served if it has been cached in the content store; otherwise, an entry is made in the PIT and the Flow Table, and the interest packet is forwarded to q_3 through n_8. The content is served if it has been cached in the content store; otherwise, an entry is made in the PIT and the Flow Table, and the interest packet is forwarded to P_1. Producer P_1 responds to the interest packet by issuing corresponding content towards C_1 through the path q_3, n_8, q_2, n_7. The first encounter of the QoS content with a QoS node q_3 invokes the admission-control algorithm for its content store, and leads to the adoption of the Flow ID for that content, so that all future content belonging to that Flow ID is cached in q_3. q_2 knows that the content Flow ID is adopted by q_3 through interest/data packet exchange triggered by the admission-control algorithm, so the content is streamed through q_2 without adoption or caching. The format of the QoS table for q_3 is shown in Table 4.

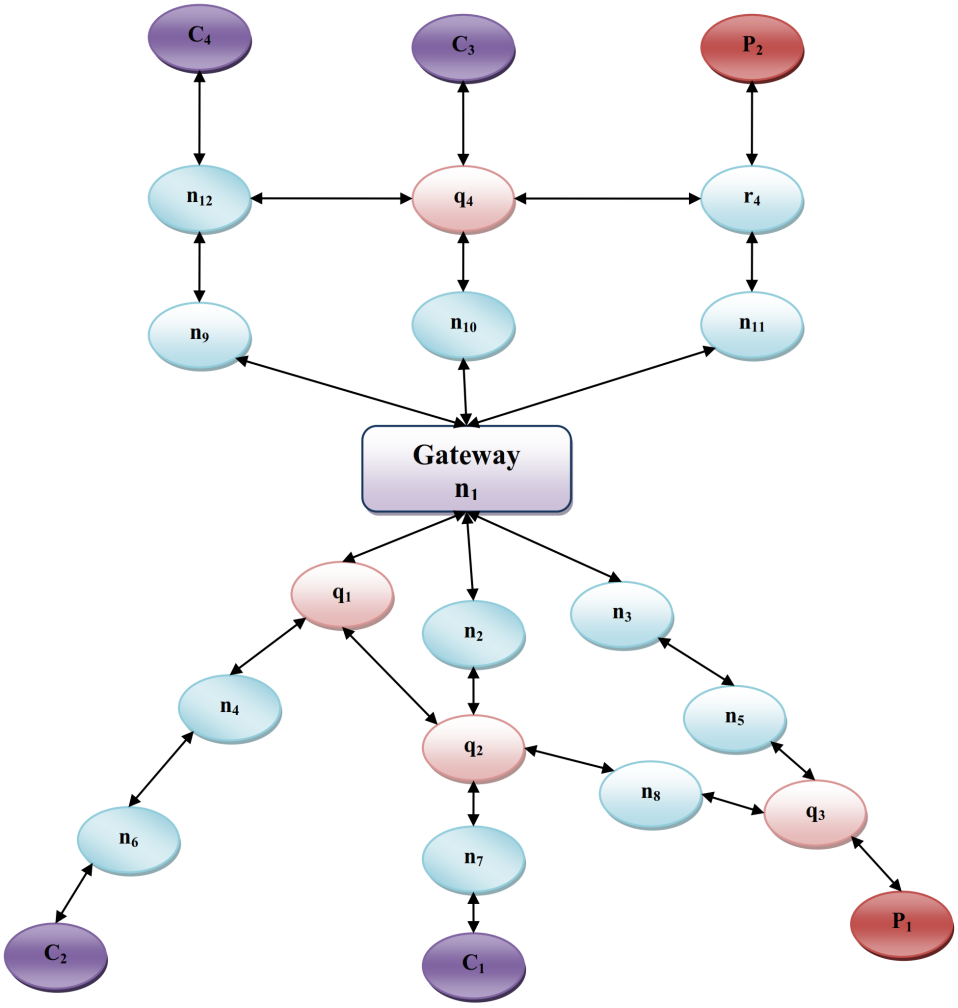


Figure 3. A representation of a path connecting content producers and consumers for illustration.

5. Results and Discussion

5.1. Simulation Setup

A QoS-linked privileged content-caching strategy is implemented using ndnSIM [30]. ndnSIM is a module of the NS-3 [41] simulator, which implements the NDN architecture for simulation purposes. ndnSIM executes as a network-layer protocol over any of the link-layer protocols (point-to-point, CSMA, etc). ndnSIM implements NDN components, i.e., PIT, FIB, CS, in a modular fashion, with interfaces for implementing new user-defined algorithms. The modular structure of ndnSIM facilitates modifications for NDN components with no/minimal effect on other modules [30].

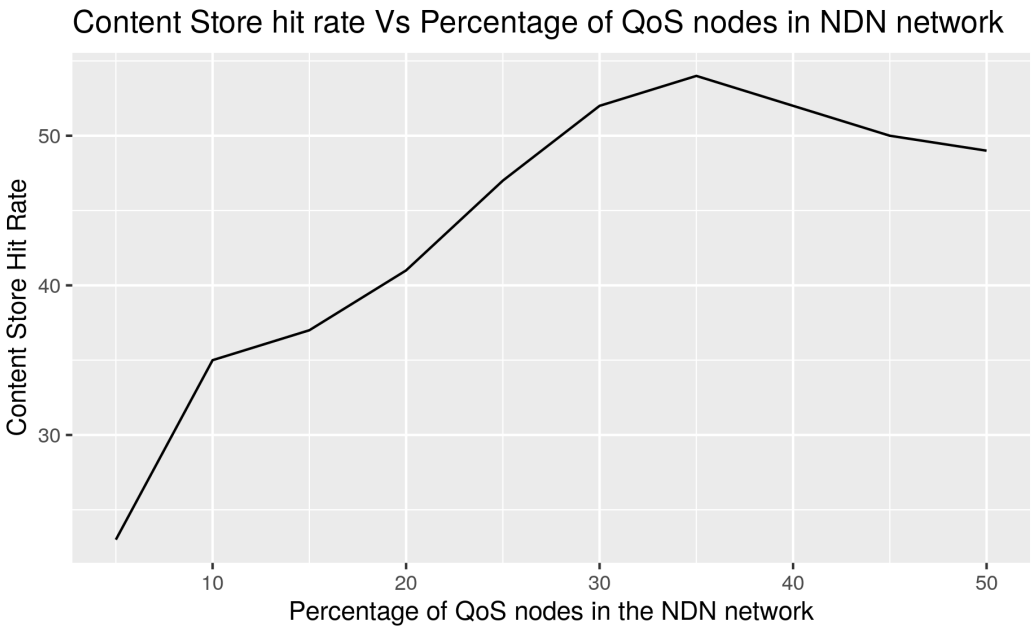
The ndnSIM is executed on a personal computer with an Intel i5 processor having 3.80 GHz clock frequency, 64 GB RAM, and Ubuntu 18.04 operating system. The total number of nodes in the NDN network is $|M|$ = 500 and $|M|$ = 1000. For a given value of $|M|$, (0.1)$|M|$ are content producers and (0.35)$|M|$ are QoS nodes as the default setting, if not mentioned otherwise, (0.25)$|M|$ are consumer nodes and (0.3)$|M|$ are intermediate NDN nodes. The 6LoWPAN protocol, along with the IEEE 802.15.4 standard, is employed for simulating a low-power IoT network scenario. The memory value of the content store is set to 10 MB for intermediate NDN nodes, with a default of 100 MB for QoS nodes if not specified otherwise. A total of 20% of the interest packets generated are for priority content [28]. The simulation scenario is executed ten times for each memory size value of the content store. The average of the outputs is plotted as a result. The results of QLPCC are compared with EQPR [28], PRR [29], probability cache, and LFU and LFF schemes, and a subsequent analysis is provided in Sections 5.2–5.5. The network settings are provided in Table 5.

Table 5. Network simulation settings.

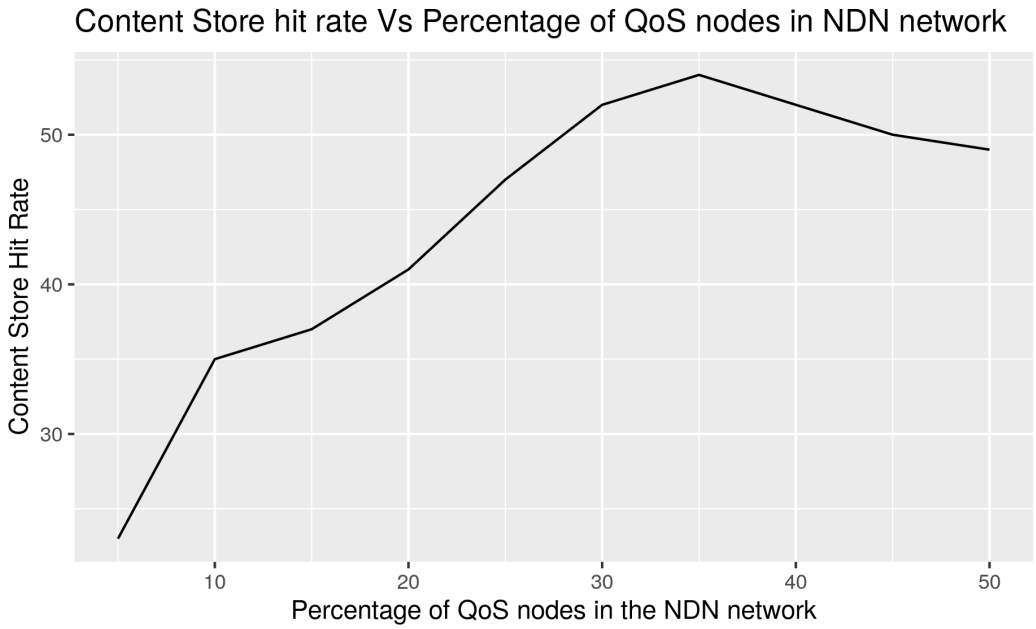
Attribute	Value
Link delay	15 milliseconds
Bandwidth	1 mbps
Routing algorithm	NDN-adaptive routing
Packet payload size	1024 bytes
Duration of simulation	4000 simulation seconds
Interest packet rate per second	$\delta = 50$ & $\delta = 100$

5.2. Content Store Hit Rate and Percentage of QoS Nodes in the Network

This evaluation answers the question "what percentage of overall NDN nodes should be designated as QoS nodes?" For $|M|$ = 500, 50 endpoint nodes are content producers. The network generates $\delta = 50$ interest packets per second towards content producers, and the QoS nodes vary between 5% and 50% in number. For $|M|$ = 1000, 100 endpoint nodes are content producers, and the network generates $\delta = 100$ interest packets per second towards content producers; the QoS nodes in the network vary between 5% and 50% in number. Figure 4 depicts the percentage of the content store hit rate, corresponding to the percentage of QoS nodes in the network. It can be observed that the content store hit rate increases till (0.35) $|M|$, i.e., 35% of the network nodes are QoS nodes, and begins to decrease because although QoS traffic is given a higher priority, regular traffic is allocated less content score space. It is important to designate the optimum number of NDN nodes as QoS nodes to obtain the desired results.



(**a**)



(**b**)

Figure 4. Content store hit rate vs. percentage of QoS nodes in the network. (**a**) Content store hit rate vs. percentage of QoS nodes in the network for $|M|$ **= 500** . (**b**) Content store hit rate vs. percentage of QoS nodes in the network for $|M|$ **= 1000**.

5.3. Content Store Size and Hit Rate

The size of a content store of intermediate NDN nodes is set at 10 MB, and the size of a content store of QoS nodes is varied between 50 MB and 250 MB. The overall average of the content store hit rate for the entire network is determined. As mentioned in the simulation setup, 20% of the traffic is priority content. Section 5.3.1 presents the evaluation of how this priority content was treated in the network, and Section 5.3.2 presents the evaluation of how the overall content was treated by the network. It can be observed that QoS traffic obtained better treatment than non-QoS traffic, when both evaluations are compared.

5.3.1. Content Store Size and Hit Rate for QoS Traffic

The simulation evaluates the content store hit rate corresponding to the size of a content store of QoS nodes for the entire network, considering only the priority content among all other traffic. QLPCC outperforms EQPR in the range of 6% to 10%, PRR in the range of 7% to 9%, the probability cache scheme in the range of 37% to 40%, the LFF scheme in the range of 41% to 45%, and the LFU scheme in the range of 44% to 47%. The results are consistent when the network nodes are scaled from $|M| = 500$ to $|M| = 1000$. The content store size of the QoS nodes ranges from 50 MB to 250 MB, as shown in the X axis of the graphs in Figure 5.

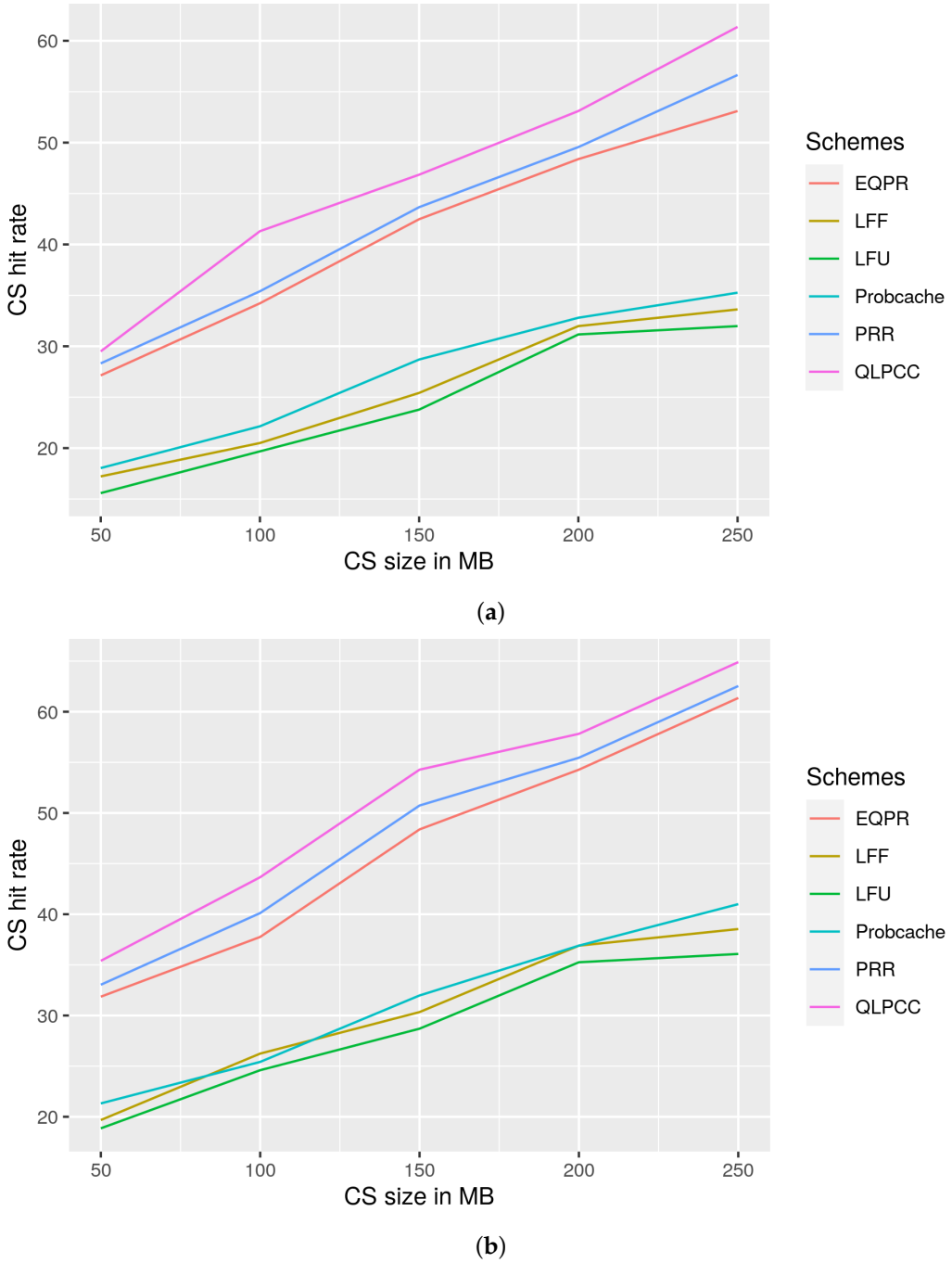


Figure 5. Content store size vs. hit rate for QoS traffic. (**a**) Content store size vs. hit rate (QoS traffic) for $|M|$ **= 500.** (**b**) Content store size vs. hit rate (QoS traffic) for $|M|$ **= 1000.**

5.3.2. Content Store Size and Hit Rate for Overall Traffic

The simulation evaluates the content store hit rate corresponding to the size of a content store size of QoS nodes for the overall traffic, including priority content (20%) and non-priority content of the entire network. The results are depicted in Figure 6, which are compared with EQPR, PRR, probability cache, and LFU and LFF schemes. QLPCC outperforms the EQPR scheme in the range of 8% to 14%, the PRR scheme in the range of 4% to 9%, the probability cache scheme in the range of 12% to 18%, the LFF scheme in the range of 16% to 22%, and the LFU scheme in the range of 24% to 26%. The results are consistent when the network nodes are scaled from $|M| = 500$ to $|M| = 1000$. The content store size ranges from 50 MB to 250 MB, as shown in the X axis of the graphs in Figure 6.

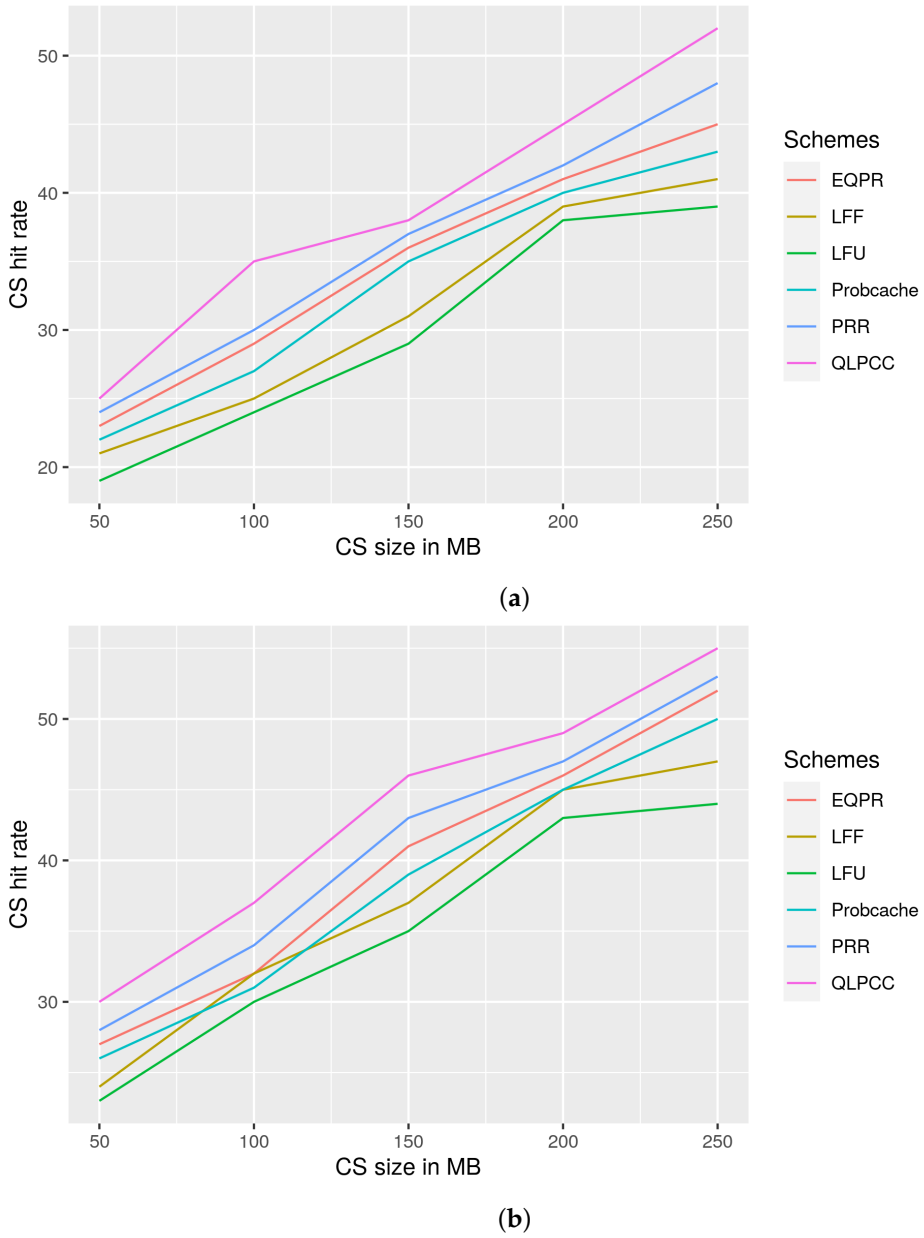


Figure 6. Content store size vs. hit rate for overall traffic. (**a**) Content store size vs. hit rate (overall traffic) for $|M|$ **= 500.** (**b**) Content store size vs. hit rate (overall traffic) for $|M|$ **= 1000.**

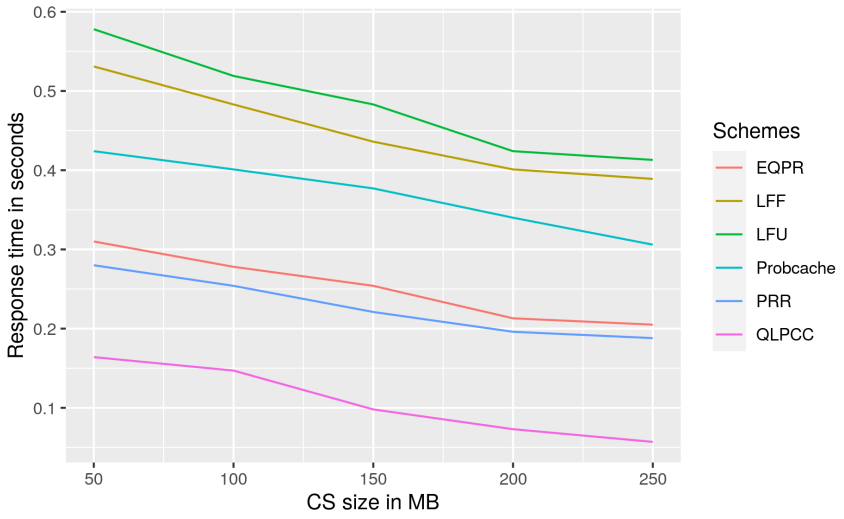
5.4. Response Time and QoS-Node Content Store Size

A lower content response time is a desirable characteristic in every network scenario, and more so in networks implementing QoS privileges. A lower response time provides a

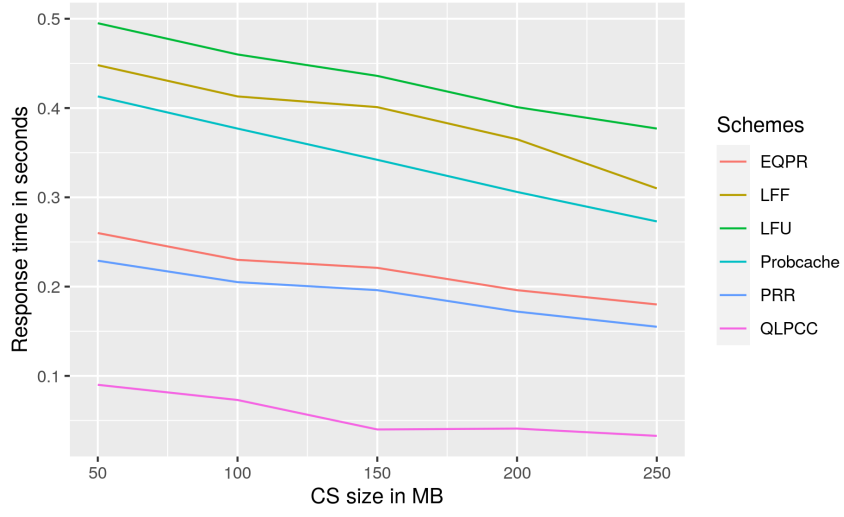
better user experience and signifies lower traffic congestion and better resource utilization. Response times corresponding to the QoS-node content store size for QoS traffic among overall traffic is provided in Section 5.4.1. Response times corresponding to QoS-node content store size for overall traffic is provided in Section 5.4.2. It may be observed from both the comparisons that QoS traffic received better treatment in the network through the QLPCC strategy.

5.4.1. Response Time and QoS-Node Content Store Size for QoS Traffic

The simulation evaluates the response time for QoS traffic corresponding to varying QoS-node content store sizes. The size of a content store of intermediate NDN nodes is set at 10 MB, and the size of a content store of QoS nodes varies between 50 MB and 250 MB. Observations from Figure 7 show that QLPCC outperforms EQPR in the range of 65% to 82%, PRR in the range of 61% to 78%, the probability cache scheme in the range of 79% to 88%, the LFF scheme in the range of 80% to 89%, and the LFU scheme in the range of 82% to 90%.



(**a**)

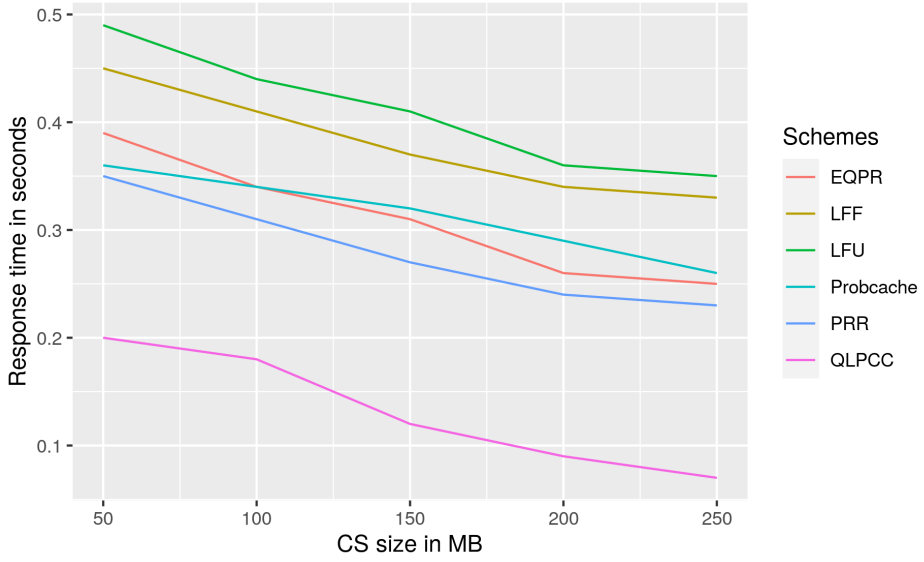


(**b**)

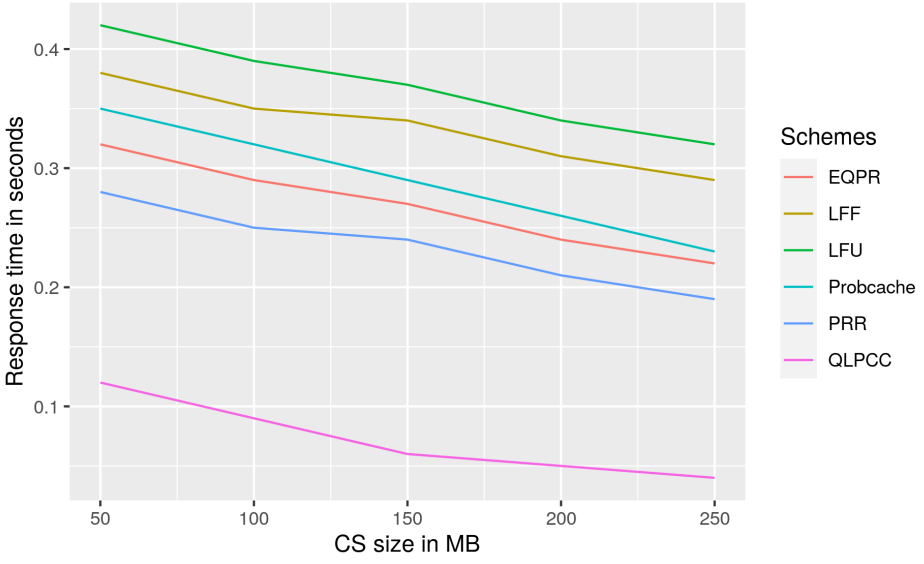
Figure 7. Response time vs. QoS-node content store size for QoS traffic. (**a**) Response time vs. QoS-node content store size (QoS traffic) for $|M|$ **= 500.** (**b**) Response time vs. QoS-node content store size (QoS traffic) for $|M|$ **= 1000.**

5.4.2. Response Time and QoS-Node Content Store Size for Overall Traffic

Response times corresponding to QoS-node content store size with interest packet rate for $|M| = 500$ and $|M| = 1000$, i.e., $\delta = 50$ and $\delta = 100$, respectively, are evaluated, and the results are plotted in Figure 8. QLPCC outperforms EQPR in the range of 49% to 72%, the PRR scheme in the range of 43% to 69%, the probability cache scheme in the range of 45% to 74%, the LFF scheme in the range of 55% to 79%, and the LFU scheme in the range of 60% to 80%. It is observed that the results are consistent when the network is scaled, proving the suitability of the scheme for network scaling. The QoS-node content store size ranges from 50 MB to 250 MB, as shown in the X axis of the graph. A graphical presentation of the results is shown in Figure 8.



(**a**)



(**b**)

Figure 8. Response time vs. QoS-node content store size for overall traffic. (**a**) Response time vs. QoS-node content store size (overall traffic) for $|M|$ **= 500.** (**b**) Response time vs. QoS-node content store size (overall traffic) for $|M|$ **= 1000.**

5.5. Hop Count and Content Store Size

The hop count signifies the number of routers that an interest packet has to travel through to fetch the corresponding data packet. A reduction in hop count signifies a lower response time, lower traffic, and lower congestion possibilities. The hop count shows the availability of content near the consumer, which is one of the overall aims of NDN, and also of information-centric networks (ICN) in general. Section 5.5.1 evaluates the hop count corresponding to the QoS-node content store size for priority traffic among overall traffics. Section 5.5.2 evaluates the hop count corresponding to QoS-node content store size for overall traffic. By comparison of both of the evaluations, it can be inferred that QoS traffic has received better treatment through the QLPCC strategy.

5.5.1. Hop Count and Content Store Size for QoS Traffic

QLPCC outperforms the EQPR scheme in the range of 60% to 64%, the PRR scheme in the range of 50% to 64%, the probability cache scheme in the range of 78% to 84%, the LFF scheme in the range of 80% to 84%, and the LFU scheme in the range of 80% to 84%. The QoS-node content store size ranges from 50 MB to 250 MB, as shown in the X axis of the graph below. A graphical presentation of the result is provided in Figure 9.

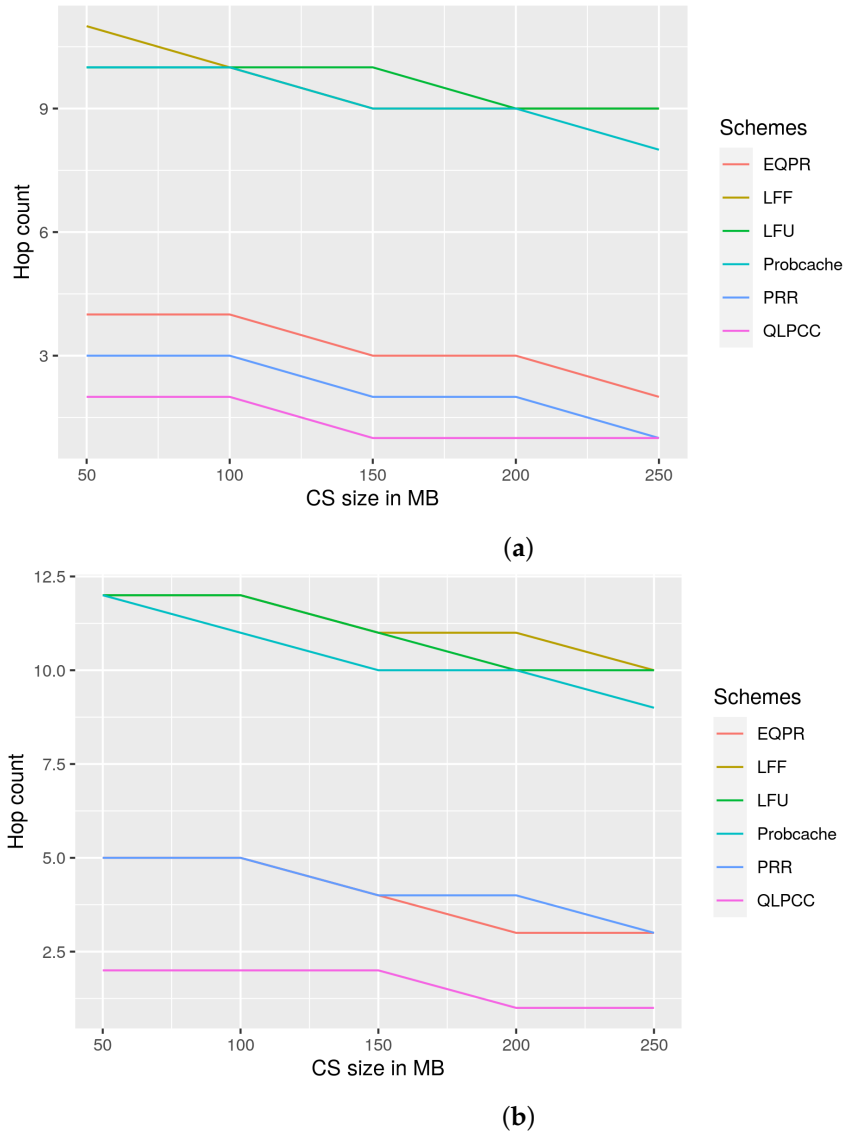
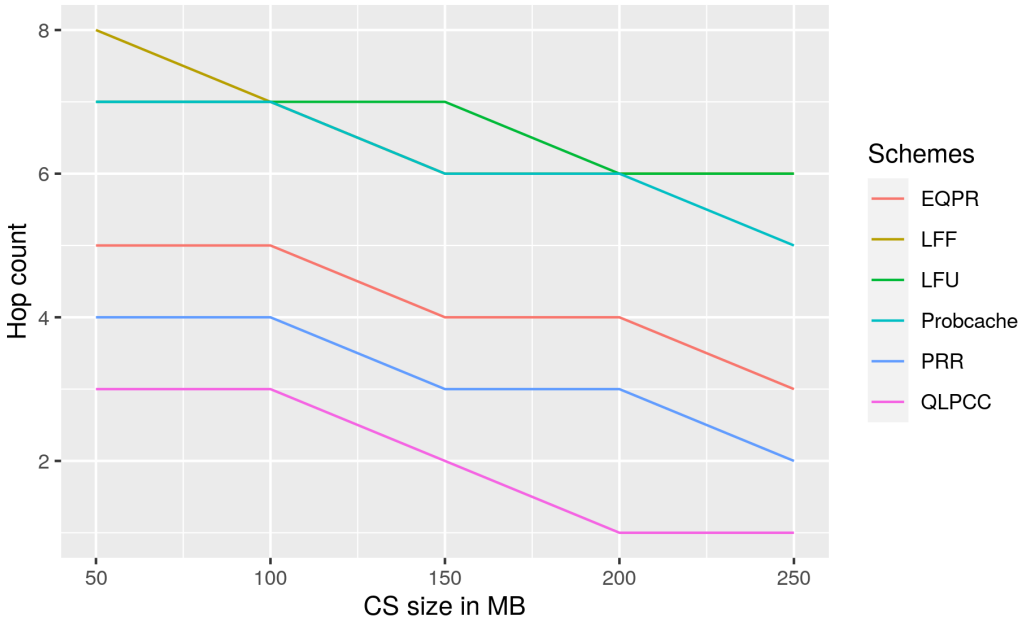


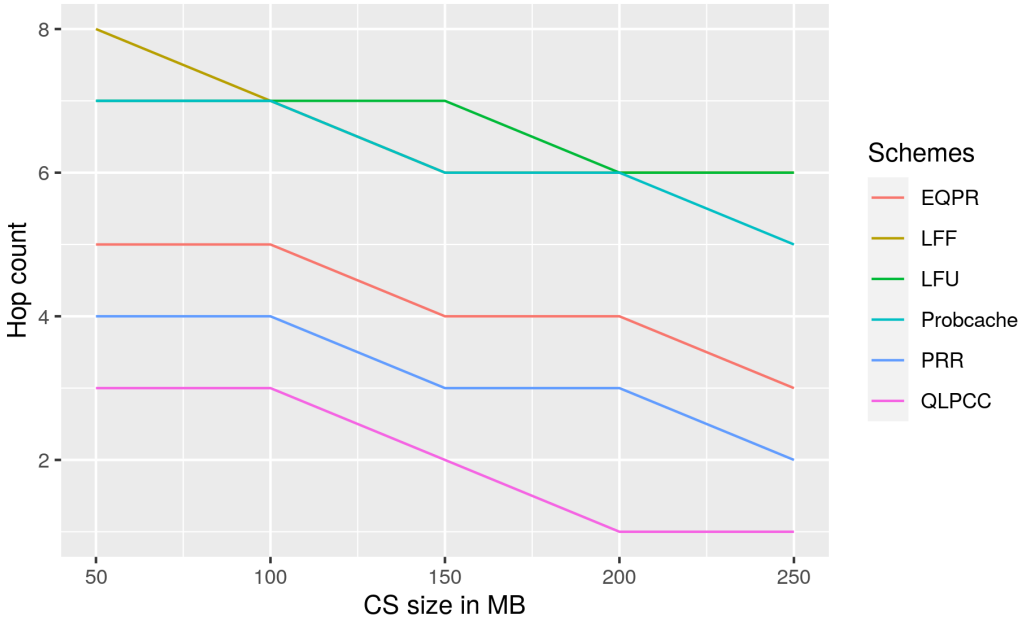
Figure 9. Hop count vs. content store size for QoS traffic. (**a**) Hop count vs. content store size (QoS traffic) for $|M|$ **= 500.** (**b**) Hop count vs. content store size (QoS traffic) for $|M|$ **= 1000.**

5.5.2. Hop Count and Content Store Size for Overall Traffic

QLPCC outperforms the EQPR scheme in the range of 57% to 60%, the PRR scheme in the range of 57% to 60%, the probability cache scheme in the range of 65% to 67%, the LFF scheme in the range of 66% to 72%, and the LFU scheme in the range of 66% to 72%. The QoS-node content store size ranges from 50 MB to 250 MB, as shown in the X axis of the graph provided in Figure 10.



(**a**)



(**b**)

Figure 10. Hop count vs. content store size for overall traffic. (**a**) Hop count vs. content store size (overall traffic) for $|M|$ **= 500.** (**b**) Hop count vs. content store size (overall traffic) for $|M|$ **= 1000.**

6. Conclusions

QoS-linked privileged content caching (QLPCC) is a Pending Interest Table (PIT) and content store (CS) management strategy for NDN-based networks for the scenarios of Quality-of-Service (QoS) implementation. QLPCC proposes a QoS node as an intermediate resourceful NDN node which handles only QoS traffic in the network. QoS traffic receives regular treatment in other NDN nodes, which ensures that regular traffic is not starved. QLPCC proposes a Flow Table-based PIT management scheme, which categorizes each QoS traffic into (prompt, reliable), prompt, reliable, and regular service categories [14–16], and calculates an eviction score for the content of the traffic identified by a Flow ID. The PIT entry with the highest eviction score is pre-empted when the need arises. QLPCC proposes a QoS Table-based content store management scheme for content-admission-control and content-eviction operations. The QoS Table maintains a record of content expiry times, Flow IDs, and paths for QoS nodes which have adopted other Flow IDs. Using QoS Tables, the QLPCC strategy decides whether to cache the content or to forward the content towards other QoS nodes. QLPCC proposes a content store eviction algorithm to free the limited memory. The content-eviction algorithm evicts the content in the order of obsolete content, prompt-privileged content, and reliable-privileged content, respectively. The proposed QLPCC strategy is simulated on an ndnSIM [30] platform for 500 nodes and 1000 nodes as a proof of scalability. The results are compared with EQPR [28], PRR [29], probability cache, and Least Frequently Used (LFU) and Least Fresh First (LFF) schemes, and are analyzed from the viewpoint of content store size vs. hit rate, response time vs. QoS-node content store size, and hop count vs. content store size. QLPCC has outperformed all of the previously mentioned measures.

For NDN-based ad hoc networks, NDN nodes should be dynamically designated as QoS nodes. Future work explores this challenge and the heuristics required to solve this problem.

Author Contributions: Conceptualization, S.H.S.; methodology, S.H.S. and U.B.; software, S.H.S.; validation, S.H.S. and U.B.; formal analysis, S.H.S.; investigation, S.H.S.; resources, U.B.; data curation, S.H.S. and U.B.; writing—original draft preparation, S.H.S.; writing—review and editing, S.H.S. and U.B.; visualization, S.H.S.; supervision, U.B.; project administration, U.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not Applicable, the study does not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

Please refer to the CRediT taxonomy for term explanations. The following abbreviations are used in this manuscript:

MDPI	Multidisciplinary Digital Publishing Institute
DOAJ	Directory of open-access journals
NDN	Named Data Networks
CS	Content store
PIT	Pending Interest Table
IOT	Internet Of Things
ICN	Information-centric networking
QoS	Quality of service
QLPCC	Quality of service-linked privileged content caching
IETF	Internet Engineering Task Force
LFF	Least Fresh First
LFU	Least Frequently Used
ECNCT	Equivalence class name component type

References

1. Javaid, M.; Haleem, A.; Pratap Singh, R.; Suman, R. Significance of Quality 4.0 towards comprehensive enhancement in manufacturing sector. *Sens. Int.* **2021**, *2*, 100109. [CrossRef]
2. Wotawa, F.; Peischl, B.; Klück, F.; Nica, M. Quality assurance methodologies for automated driving. *Elektrotech. Informationstechnik* **2018**, *135*, 322–327. [CrossRef]
3. Sallum, E.; Pereira, N.; Alves, M.; Santos, M. Improving Quality-Of-Service in LoRa Low-Power Wide-Area Networks through Optimized Radio Resource Management. *J. Sens. Actuator Netw.* **2020**, *9*, 10. [CrossRef]
4. Zhang, L.; Afanasyev, A.; Burke, J.; Jacobson, V.; Claffy, K.; Crowley, P.; Papadopoulos, C.; Wang, L.; Zhang, B. Named Data Networking. *SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 66–73. [CrossRef]
5. Touati, H.; Aboud, A.; Hnich, B. Named Data Networking-based communication model for Internet of Things using energy aware forwarding strategy and smart sleep mode. *Concurr. Comput. Pract. Exp.* **2021**, *34*, e6584. [CrossRef]
6. V, A.; Kiran, A.G. SynthNet: A skip connected depthwise separable neural network for Novel View Synthesis of solid objects. *Results Eng.* **2022**, *13*, 100383. [CrossRef]
7. Shrisha, H.; Boregowda, U. An energy efficient and scalable endpoint linked green content caching for Named Data Network based Internet of Things. *Results Eng.* **2022**, *13*, 100345. [CrossRef]
8. Gündoğan, C.; Kietzmann, P.; Lenders, M.; Petersen, H.; Schmidt, T.C.; Wählisch, M. NDN, CoAP, and MQTT: A Comparative Measurement Study in the IoT. In Proceedings of the 5th ACM Conference on Information-Centric Networking, ICN '18, Boston, MA, USA, 21–23 September 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 159–171. [CrossRef]
9. Chakraborti, A.; Amin, S.O.; Azgin, A.; Misra, S.; Ravindran, R. Using ICN Slicing Framework to Build an IoT Edge Network. In Proceedings of the 5th ACM Conference on Information-Centric Networking, ICN '18, Boston, MA, USA, 21–23 September 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 214–215. [CrossRef]
10. Karamchati, S.; Rawat, S.; Yarram, S.; Ramaguru, G.P. Mapping mechanism to enhance QoS in IP networks. In Proceedings of the 2018 International Conference on Information Networking (ICOIN), Chiang Mai, Thailand, 10–12 January 2018; pp. 797–803. [CrossRef]
11. Chen, X.; Li, Z.; Zhang, Y.; Long, R.; Yu, H.; Du, X.; Guizani, M. Reinforcement learning–based QoS/QoE-aware service function chaining in software-driven 5G slices. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3477. [CrossRef]
12. Gündoğan, C.; Pfender, J.; Kietzmann, P.; Schmidt, T.C.; Wählisch, M. On the impact of QoS management in an Information-centric Internet of Things. *Comput. Commun.* **2020**, *154*, 160–172. [CrossRef]
13. Yu, Z.; Hu, J.; Min, G.; Lu, H.; Zhao, Z.; Wang, H.; Georgalas, N. Federated Learning Based Proactive Content Caching in Edge Computing. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6. [CrossRef]
14. Flow Classification in Information Centric Networking, Internet-Draft—Work in Progress 05. Available online: https://tools.ietf.org/id/draft-moiseenko-icnrg-flowclass-06.html (accessed on 6 January 2022).
15. Oran, D.R. *Considerations in the Development of a QoS Architecture for CCNx-like ICN Protocols. Internet-Draft Draft-Oran-Icnrg-Qosarch-02, Internet Engineering Task Force*; 2020; Work in Progress. Available online: http://www.watersprings.org/pub/id/draft-oran-icnrg-qosarch-03.html (accessed on 2 May 2022).
16. Gündogan, C.; Schmidt, T.C.; Wählisch, M.; Frey, M.; Shzu-Juraschek, F.; Pfender, J. *Quality of Service for ICN in the IoT. Internet-Draft draft-gundogan-icnrg-iotqos-01, Internet Engineering Task Force*; 2019; Work in Progress. Available online: https://datatracker.ietf.org/meeting/interim-2019-icnrg-04/materials/slides-interim-2019-icnrg-04-sessa-qos-for-icn-in-the-iot-00 (accessed on 2 May 2022).
17. Zheng, Q.; Kan, Y.; Chen, J.; Wang, S.; Tian, H. A Cache Replication Strategy Based on Betweenness and Edge Popularity in Named Data Networking. In Proceedings of the ICC 2019—2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7. [CrossRef]
18. Kamiyama, N.; Murata, M. Dispersing Content Over Networks in Information-Centric Networking. *IEEE Trans. Netw. Serv. Manag.* **2019**, *16*, 521–534. [CrossRef]
19. Im, Y.; Prahladan, P.; Kim, T.H.; Hong, Y.G.; Ha, S. SNN-cache: A practical machine learning-based caching system utilizing the inter-relationships of requests. In Proceedings of the 2018 52nd Annual Conference on Information Sciences and Systems (CISS), Princeton, NJ, USA, 21–23 March 2018; pp. 1–6. [CrossRef]
20. Zhang, R.; Liu, J.; Huang, T.; Xie, R. Popularity based probabilistic caching strategy design for named data networking. In Proceedings of the 2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Atlanta, GA, USA, 1–4 May 2017; pp. 476–481. [CrossRef]
21. Xu, C.; Quan, W.; Zhang, H.; Grieco, L.A. GrIMS: Green Information-Centric Multimedia Streaming Framework in Vehicular Ad Hoc Networks. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 483–498. [CrossRef]
22. Wang, L.; Tyson, G.; Kangasharju, J.; Crowcroft, J. Milking the Cache Cow with Fairness in Mind. *IEEE/ACM Trans. Netw.* **2017**, *25*, 2686–2700. [CrossRef]
23. Wang, S.; Bi, J.; Wu, J.; Vasilakos, A.V. CPHR: In-Network Caching for Information-Centric Networking with Partitioning and Hash-Routing. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2742–2755. [CrossRef]

24. Alubady, R.; Hassan, S.; Habbal, A. HLLR: Highest Lifetime Least Request policy for high performance Pending Interest Table. In Proceedings of the 2016 IEEE Conference on Open Systems (ICOS), Langkawi, Malaysia, 10–12 October 2016; pp. 42–47. [CrossRef]
25. Ryu, S.; Joe, I.; Kim, W. Intelligent Forwarding Strategy for Congestion Control Using Q-Learning and LSTM in Named Data Networking. *Mob. Inf. Syst.* **2021**, *2021*, 5595260. [CrossRef]
26. Han, B.; Tao, Y.; Zhu, Y. A PIT Replacement Policy Based on Minimum Contribution in NDN-IoT. In Proceedings of the 2019 2nd International Conference on Hot Information-Centric Networking (HotICN), Chongqing, China, 13–15 December 2019; pp. 13–17. [CrossRef]
27. Manisha, G.; Emil Selvan, G.S.R.; Ramkumar, M.P. Pending Interest Lifetime Mechanism for Vehicular Named Data Networks. In Proceedings of the 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 30–31 March 2019; pp. 1–6. [CrossRef]
28. Buragohain, M.; Gudipudi, P.; Anwer, M.Z.; Nandi, S. EQPR: Enhancing QoS in Named Data Networking using Priority and RTT Driven PIT Replacement Policy. In Proceedings of the ICC 2019—2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7. [CrossRef]
29. Buragohain, M.; Nandi, S. Quality of Service provisioning in Named Data Networking via PIT entry reservation and PIT replacement policy. *Comput. Commun.* **2020**, *155*, 166–183. [CrossRef]
30. Mastorakis, S.; Afanasyev, A.; Moiseenko, I.; Zhang, L. *ndnSIM 2: An Updated NDN Simulator for NS-3*; Technical Report NDN-0028, Revision 2; NDN: 2016. Available online: https://irl.cs.ucla.edu/data/files/techreports/ndn0028-2.pdf (accessed on 2 May 2022).
31. Wang, X.; Lu, Y. Efficient Forwarding and Data Acquisition in NDN-based MANET. *IEEE Trans. Mob. Comput.* **2020**, *21*, 530–539. [CrossRef]
32. Lv, J.; Wang, X.; Li, Q.; Pan, T. Caching-enabled networks and its applications into the emerging networking paradigms and technologies. *Internet Technol. Lett.* **2021**, *4*, e316. [CrossRef]
33. Amadeo, M.; Campolo, C.; Molinaro, A. Named data networking for priority-based content dissemination in VANETs. In Proceedings of the 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Valencia, Spain, 4–8 September 2016; pp. 1–6. [CrossRef]
34. Meddeb, M.; Dhraief, A.; Belghith, A.; Monteil, T.; Drira, K.; AlAhmadi, S. Cache Freshness in Named Data Networking for the Internet of Things. *Comput. J.* **2018**, *61*, 1496–1511. [CrossRef]
35. Pfender, J.; Valera, A.; Seah, W.K. Easy as ABC: A Lightweight Centrality-Based Caching Strategy for Information-Centric IoT. In Proceedings of the 6th ACM Conference on Information-Centric Networking, ICN '19, Macao, China, 24–26 September 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 100–111. [CrossRef]
36. Ribeiro, A.V.; Sampaio, L.N.; Ziviani, A. Affinity-Based User Clustering for Efficient Edge Caching in Content-Centric Cellular Networks. In Proceedings of the 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, 25–28 June 2018; pp. 474–479. [CrossRef]
37. Zeng, Y.; Hong, X. *A Caching Strategy in Mobile Ad Hoc Named Data Network*; IEEE: Piscataway, NJ, USA, 2012. [CrossRef]
38. Pfender, J.; Valera, A.; Seah, W.K.G. Performance Comparison of Caching Strategies for Information-Centric IoT. In Proceedings of the 5th ACM Conference on Information-Centric Networking, ICN '18, Boston, MA, USA, 21–23 September 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 43–53. [CrossRef]
39. Wu, H.; Li, J.; Zhi, J.; Ren, Y.; Li, L. Design and Evaluation of Probabilistic Caching in Information-Centric Networking. *IEEE Access* **2018**, *6*, 32754–32768. [CrossRef]
40. Khan, J.A.; Westphal, C.; Garcia-Luna-Aceves, J.J.; Ghamri-Doudane, Y. NICE: Network-Oriented Information-Centric Centrality for Efficiency in Cache Management. In Proceedings of the 5th ACM Conference on Information-Centric Networking, ICN '18, Boston, MA, USA, 21–23 September 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 31–42. [CrossRef]
41. Riley, G.F.; Henderson, T.R. The ns-3 Network Simulator. In *Modeling and Tools for Network Simulation*; Wehrle, K., Günes, M., Gross, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 15–34.