Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 8358438, 18 pages
https://doi.org/10.1155/2023/8358438

Research Article

PoSiF: A Transient Content Caching and Replacement Scheme for ICN-IoT

Geetu Dhawan, Arka P. Mazumdar, and Yogesh K. Meena

Malaviya National Institute of Technology, Jaipur 302017, Rajasthan, India

Correspondence should be addressed to Arka P. Mazumdar; apmazumdar.cse@mnit.ac.in

Received 18 February 2023; Revised 23 August 2023; Accepted 29 September 2023; Published 29 November 2023

Academic Editor: SK Hafizul Islam

Efficient data management plays a critical role in facilitating smooth communication and optimizing resource utilization within the rapidly growing ecosystem of the internet of things (IoT), which creates vast amounts of content. The emergence of Information-Centric Networking (ICN) has positioned it as a promising paradigm for facilitating IoT applications, as it prioritizes content-centric communication. This inherent characteristic of ICN has the potential to greatly improve the performance of IoT systems through the reduction of latency, conservation of network capacity, and enhancement of scalability. Existing research in this area focuses primarily on the popularity and freshness of the contents. However, they overlook the limited cache size of constraint devices, which should be utilized efficiently. This article presents a popularity, size, and freshness-based (PoSiF) transient content caching, and cache replacement scheme for ICN-enabled IoT devices. For content placement and replacement, the PoSiF scheme aims to enhance cache utilization through these three parameters while addressing the tradeoff between content size and popularity. The PoSiF technique demonstrates a significant enhancement in cache hit rates, with an improvement of 21.6% compared to the existing CCS/CES scheme, while effectively reducing hop ratios by 42.5%. On the other hand, cache replacement observes improved cache hits of 18% over least recently used (LRU) and 7.2% over least frequently used (LFU) while decreasing the average distance by 4.1% and 7.2%, respectively. While the retrieved freshness is 8.12% and 3.2% less than the LRU and LFU schemes, PoSiF compensates by satisfying more queries from cache-based content storage than the alternative methods.

1. Introduction

The growing and pervasive use of the internet of things (IoT) affects every aspect of life. Internet traffic has increased dramatically over the past few years as internet users have increased exponentially. Due to this exponential growth, the data tsunami was well-anticipated. Video and mobile traffic are also anticipated to significantly contribute to this increased volume of devices and traffic [1]. However, this growth will place additional restrictions on IP-based networks and raise numerous issues, such as scalability, energy consumption, and network congestion [2, 3]. Moreover, the evolution of new technologies has spawned new requirements that pose challenges to the existing client-server architecture in terms of supporting scalable content distribution, mobility, and security, among others [4–6]. To address these issues, several paradigms, such as content delivery and security patches, are proposed for the existing architecture, some of which increase the network's complexity and create the issue of patching over patches.

Information-Centric Networking (ICN) has recently gained attraction as a viable candidate for the future architecture of the internet [7, 8], where it intends to replace the address-based internet architecture with a named content-based architecture. While feasibility and design considerations for applying ICN to IoT are debated [9] in the literature, in-network caching is an essential feature of ICN that makes the availability of content closer to the requester [10, 11]. The typical size of IoT content is small; even a tiny cache of IoT devices can hold many IoT packets [12, 13]. In addition, IoT devices are limited in terms of memory, energy, and bandwidth, making ICN an attractive alternative. Figure 1 illustrates the fundamental design of ICN-based IoT, in which IoT nodes publish the content and subscribers express interest in retrieving the content from the device cache. In order to alleviate the load on the

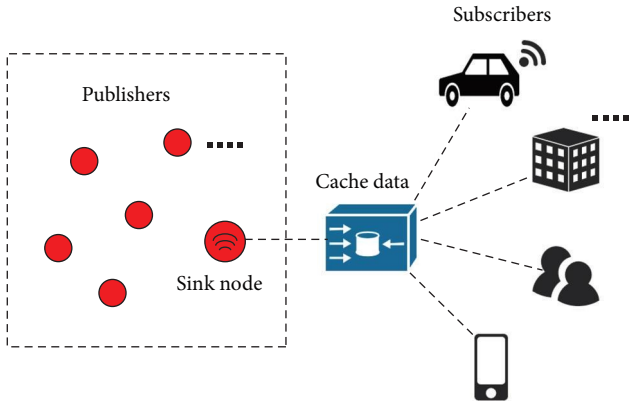

Figure 1: ICN-IoT network: an example.

network between the publisher and the subscriber, integration of ICN into the IoT may assist.

Caching transient content introduces a novel challenge compared to the caching static or relatively stable content. It is challenging to manage content that is dynamic, transient, and subject to rapidly changing popularity and access patterns. Existing research in this area focuses on the popularity and freshness as thresholds toward content selection for caching, disregarding the limited cache size, and the relationship between content freshness and cache parameters. This work aims to present a novel approach that considers content size and popularity to deduce the content's weight for both caching and replacement. We propose an IoT caching and replacement scheme, PoSiF, that considers essential IoT parameters, including content popularity, content freshness, content size, and cache size, while making caching decisions. The primary contributions can be summarized as follows:

(i) The proposed scheme takes into account the content's size, popularity, and freshness in order to leverage the cache more effectively, as constrained devices may have limited cache capacities.

(ii) To maximize caching efficiency, the proposed scheme employs a tradeoff between content size and popularity.

(iii) Since replacement decisions should be influenced by the IoT parameters, it finally proposes a cache replacement scheme that aims to replace the contents of the cache that have lower weights.

The rest of the article is organized as follows: Section 2 provides an overview of ICN architectures and ICN for IoT and a concise literature review of the relevant articles. While the proposed PoSiF scheme is presented in Section 3, the experimental setup, simulation results, and analysis are presented in Section 4. The article concludes with Section 5.

2. Background and Literature Review

This section details the necessary features to be present in any ICN strategy. It first focuses on the fundamentals of the technology, such as the various terminologies used in ICN, the different ICN models, and the significance of ICN in IoT. Finally, it presents a brief literature review in this context.

2.1. Background. ICN can be compatible with the IoT, creating new opportunities by addressing content by its name. Literature has demonstrated that ICN-IoT can outperform [23] IP-based IoT due to its many features and benefits, such as naming, caching, and security. In ICN, content is dispersed and may already be cached in a local node; therefore, retrieving it from the content source is only sometimes necessary. As a result, this ICN-based strategy may serve as a viable alternative for intermittently connected nodes [12, 19–21]. The importance of caching in such contexts is further illustrated by a fog-based caching system presented in [22] for IoT applications that use the ICN architecture. In the literature, ICN architectures such as named data networking (NDN) [17], publisher-subscriber internet technology (PURSUIT) [16], convergence [4], scalable and adaptive internet solutions (SAIL), and data-oriented network architecture (DONA) [15] have been proposed. NDN differs from the other designs by incorporating stateful forwarding, content packets signed by the producer, and content names directly integrated into packet forwarding. It also supports on-path and off-path caching, thus outperforming content-centric networking (CCN).

ICN nodes, in general, maintain three data structures, namely: content store (CS), forwarding information base (FIB), and pending interest table (PIT). As depicted in Figure 2, a request is forwarded to the publisher node via intermediate nodes. These intermediate nodes maintain a CS containing cached content in order to satisfy the future demands. Conversely, the FIB stores the interface identifier and next-hop information for each reachable destination network prefix and the interface to which the interest message must be forwarded. The third data structure, PIT, records interest packets received but not responded to. The request eventually reaches the publisher, who responds with the content, and the PIT entries in the intermediate nodes for that request are deleted as it propagates back to the requester [18].

IoT applications are fundamentally content-centric; regardless of source, they are primarily concerned with the content. Every time a publisher creates new content, it is cached in the network and given a unique name identifier, irrespective of location. Users who need this content forward a request to the network, while a caching node sends the content via the shortest route to the requester [18]. Therefore, ICN in-network caching may significantly reduce the workload of content producers that receive a substantial number of requests. Distributed caching, decoupled publisher-subscriber relationships, and location-independent content delivery can significantly benefit ICN-IoT and enhance the overall performance of the potential applications [23].

2.2. Literature Review. Numerous caching strategies have been proposed for ICN, including the probabilistic in-network caching scheme, leave copy down (LCD), cache less for more, collaborative caching, and age-based caching [10, 24–28]. These strategies are meant for regularly requested, prerecorded multimedia files like YouTube videos, or other substantial

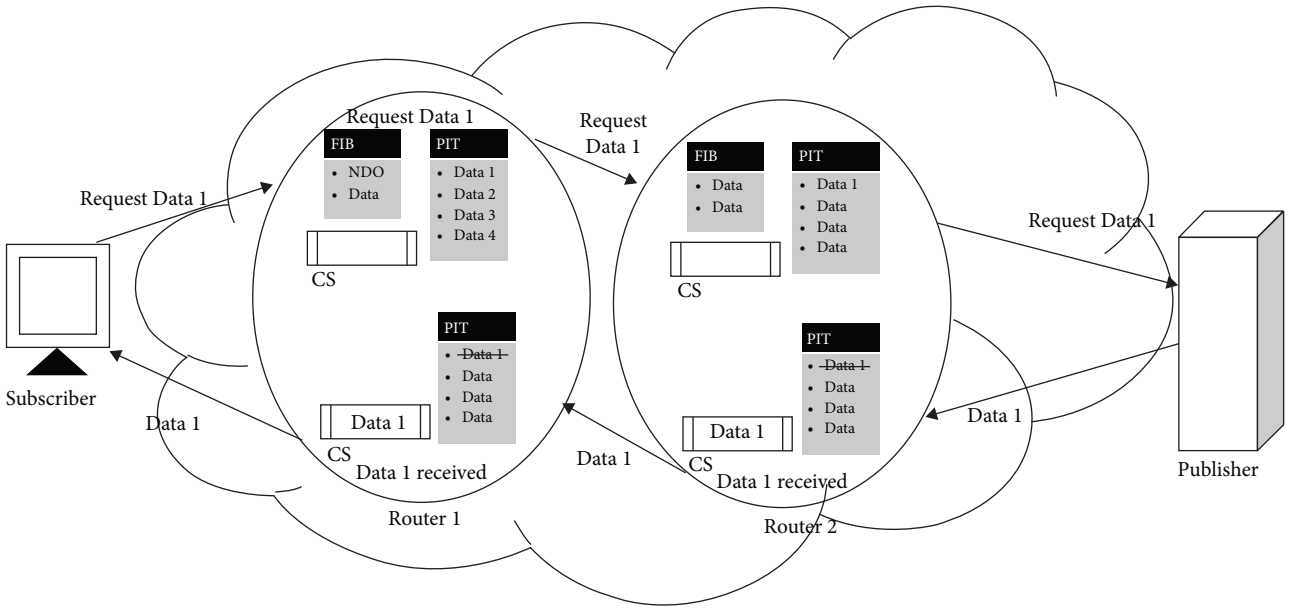

Figure 2: Named data networking: an illustrative example.

content. They were created specifically for an ICN infrastructure. IoT systems cannot directly use ICN caching techniques, and the NDN in-network caching capability, including its parameters, needs to be thoroughly investigated. This section examines the literature on ICN-IoT caching techniques; Table 1 summarizes the work. Here, we consider several factors, including approaches for caching the content, prominent parameters that were taken into account, and the drawbacks of these schemes.

IoT systems must overcome several challenges, including scalability, device heterogeneity, traffic volume, bandwidth limitations, and low-residual energy [2, 13]. Few studies have examined the deployment of NDN-IoT and highlighted the benefits and solutions that can be achieved through NDN-IoT integration. With its support for named content, ICN provides content-centric solutions to the problems of high-data volume [36], scalability, and heterogeneity [37] that have previously been identified and discussed. The work presented in [12] is the first experimental investigation of NDN in IoT and analyzes NDN's limitations in that context. To evaluate the impact of caching, the authors compared the performance of content retrieval from the various consumers with and without conventional NDN in-network caching.

The impact of the freshness of IoT content on NDN caching is discussed in [29]. In addition to the freshness parameter included in content packets, the authors propose a consumer-driven freshness approach. This mechanism enables the consumer to specify their particular requirement regarding the freshness of the interest; consequently, the caching node can determine whether it has the relevant information. The scheme's primary flaw is that it ignores important factors that could influence the decision to cache content, such as its popularity and size, and thus fails to achieve higher cache hits.

In [38], the caching of IoT content on internet content routers is described. In this scheme, the author considers freshness, and routers dynamically update their caching probability based on their hop distances to the source of incoming requests and the freshness requirement. In addition, the work defines a utility function that addresses a tradeoff between the multihop delivery from consumer to producer and the expected loss of freshness for the delivered content. The main flaw of this solution is that it ignores the content size parameter. To cache transient content, authors in [32] consider temporal properties such as content lifetime, freshness, and popularity while making caching decisions. The analysis presented by the authors is based on the tradeoff between the communication costs of content retrieval and the freshness loss cost of cached items.

IoT devices are energy constrained, and few authors discuss the need to consider residual energy while making caching decisions. The work presented in [31] demonstrates that device energy is an essential criterion for caching decisions because nodes with the sufficient energy can serve more requests in IoT devices; consequently, the authors propose a probabilistic caching strategy. The scheme considers the content freshness and the device's residual energy and storage capacity. In [30], an estimation of caching time on the next-hop node is presented. Based on a caching time model, the authors propose a cache-aware named-content forwarding scheme for weak IoT devices. The main disadvantage of this scheme is that it does not take into account the freshness and popularity of content when estimating the content caching time on neighbor nodes, which does not necessarily relate to the constrained nature of the IoT devices.

The impact of the various well-known caching and replacement policies are studied in [39]. The authors suggest appropriate caching and replacement policies for IoT networks in this article. The article examined several IoT

Table 1: Comparison summary of ICN-IoT caching approach.

Paper	Approach	Parameter	Drawback
ICN in the IoT-NDN Wild [12]	Random	No parameter considered	Does not consider any parameters
Consumer-driven information freshness [29]	Random	Freshness	Other essential IoT parameters are not considered
Cache aware NDN in IoT [30]	Probabilistic	Freshness with cache time	Not taking content's popularity into account
Probabilistic caching strategy for the internet of things [31]	Probabilistic	content freshness, energy level, and storage capability	Caching utility function may return a value even if one or more of its parameters are obsolete
Caching transient data [32]	Probability-based caching	Freshness and content popularity	Disregard content size
Lifetime-based cooperative caching (LCC) [33]	Cooperative caching scheme	Content popularity and freshness	Ignore the content and cache size parameters
DRL [34]	Deep neural network based	Content popularity and freshness	Cache size and content size parameters not considered
PUC [35]	Clustering	Content popularity and freshness	Content size not taken into account
CCS/CES [14]	Exponential weighted moving average (EWMA) and probabilistic	Freshness and popularity	Disregard essential IoT parameters like limited cache and content size not taken into account

caching and replacement strategies and offered suggestions and directions for the future work and suitable criteria to consider while caching IoT content.

The authors in [33] describe a cooperative caching scheme for ICN-IoT networks based on the lifetime of IoT content to save energy and enhance the energy efficiency of IoT devices. The intermediate nodes perform cooperative caching based on this information. The result demonstrates a reduction in overall energy consumption and the average number of hops when using the scheme. However, the scheme ignores the content and cache size parameters.

The lifetime-based cooperative caching scheme (LCC) [33] takes into account only the content lifetime and the content request rate, while other essential parameters such as cache size and content size of IoT devices are ignored. A similar approach is followed in [40], providing insights into the content's popularity and residual lifetime (time remaining before the content becomes redundant); however, it disregards the cache size and content size parameters. Another study proposes a deep reinforcement learning-based approach [34] to solve the problem of caching IoT content at the edge without knowing the popularity of IoT content in the future, user request patterns, and other context characteristics based on the content metrics. The caching agent receives multiple signals containing user requests, context characteristics, and network conditions to monitor the state of the environment. The caching agent selects a caching action based on the output and observes the reward to train and improve the deep NN model. The technique, however, does not take into account the content size or the cache size of the constrained devices.

In order to maximize the cache hit ratio, the authors in [41] proposed a scheme that caches the most popular content with the longest residual lifetime. The Packet Update Caching (PUC) scheme [35], on the other hand, employs a circular buffer to store incoming content and two distinct pointers for writing and reading operations. Both approaches overlook the importance of IoT content size, which can influence the decision to cache.

Similarly, the work presented in [14] proposed a freshness and popularity-aware caching scheme for an NDN-based IoT scenario. The author of this work proposed two important concepts: caching at the core strategy (CCS) and caching at the edge strategy (CES). CES employs popularity to make the caching decisions and freshness and popularity to devise the replacement policy. Another approach for determining the optimal node for cache placement is proposed in the CNCP scheme [45]. The approach proposes a bloom filter-based caching and content searching technique that balances cache occupancy, node energy, and content freshness. However, it does not regard the cache replacement decisions based on the tradeoff between parameters. Even though freshness, popularity, and energy are all taken into account, none of these approaches focus on the tradeoff between content size, popularity, and the size of the cache.

Due to limited cache capacity, the caching node must be discerning when deciding which content to cache. Similarly, small contents with a comparable or higher request rate fill the cache slowly and can achieve greater cache hits.

3. Proposed Work

Due to the limited resources of IoT nodes and their inability to store large amounts of content in memory, it is difficult to apply traditional ICN caching strategies to IoT content. Caching techniques that take into account key factors are required to boost performance by reducing hops and increasing cache hits. Caching transient content instead of repeatedly fetching it from the provider can save bandwidth. IoT nodes are often deployed in the remote locations with unreliable network connections and intermittent connectivity with the other communicating nodes. Moving content closer to its source enables edge nodes to provide content and decreases the load on central nodes.

In applications where sensor nodes update content periodically, older parameter values expire and become invalid. Here, the lifetime of the content is the period between two successive changes. Typically, transient content has a fixed lifetime and is regularly updated. Several transient content caching schemes have been proposed in the literature to enable this functionality [32, 38], and many others discuss caching in IoT devices [22, 31, 33–35]. The present cache replacement algorithms, however, are ineffective since the requesting node must maximize cache freshness while the contents receive updates because transient content is periodically updated. Moreover, limited cache availability in IoT nodes necessitates the retention of the most popular contents during cache replacement. This section presents the proposed PoSiF transient content caching and cache replacement scheme that considers popularity, size, and freshness while making caching decisions.

3.1. Problem Formulation. We propose a caching and replacement technique based on the content popularity, size, and freshness, with the dispersal of caching decisions depending on the tradeoff between content size and popularity. The objective is to replace cached content when it is full or has expired in order to prevent stale or expired content from being served from the cache. The following considerations must be taken into account when designing the scheme:

(i) The IoT content routers' remaining cache size may be insufficient to store all of the content along its path, resulting in cache-miss of frequently requested content.

(ii) Content request rates (or popularity) and content sizes may vary. Caching policies should prioritize smaller content with high request rates over larger content with similar or lower request rates. Otherwise, this will lead to unwarranted content store flooding and decreased cache hits.

(iii) Maintaining the content freshness of cached content is also challenging due to the fact that transient content in the content store will eventually become obsolete and invalid.

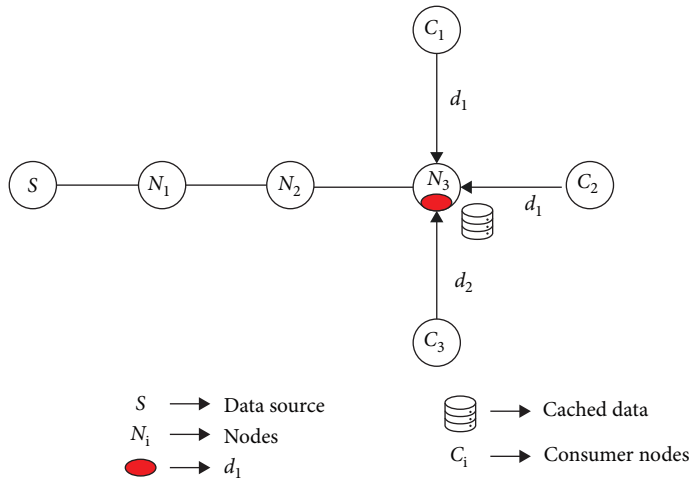

FIGURE 3: Caching a popular content in ICN-ioT.

The content producer node indicated as S, is responsible for creating content. The intermediate/caching nodes are represented by $N_1, N_2, N_3, \ldots, N_n$, while the contents are represented as $d_1, d_2, d_3, \ldots, d_n$. The consumer nodes, C_1, $C_2, C_3, \ldots, C_n$, request content from the producer or caching nodes.

3.2. Key Factors. Next, we will investigate the key factors that are considered by our proposed policy and how they influence caching decisions on the constrained devices.

3.2.1. Cache Size. We can infer that nodes with large storage capacities are more likely to cache incoming content. On the other hand, the nodes with limited storage should be very picky about what content they decide to cache. The cache size is heavily confined by the typical cache size of constrained devices, which is measured in kilobytes. Therefore, to enhance cache management for IoT devices, we take the remaining cache size into account while making caching decisions for the incoming content. Typically, sensor content is small and only a few bytes long. These contents should be updated or replaced when they reach the end of their useful life in order to make room for a new version of other data in the cache. Thus, taking cache size into account may help you to decide more wisely to increase cache utilization.

3.2.2. Content Popularity. The number of requests for specific content in a given time frame is referred to as content popularity. The popularity of content plays a vital role as a small percentage of content is requested more frequently than the others [10]. Therefore, storing popular content on devices with a small cache capacity may result in an increase in cache hits and a decrease in average hop counts. In selective caching schemes that consider content popularity, content with higher request rates should take precedence over less popular content. To illustrate, as shown in Figure 3, node N_3 receives numerous requests for content d_1 from consumers C_1 and C_2. As a result, node N_3 treats d_1 as popular content and prioritizes caching it above d_2, which only received a single request from C_3.

3.2.3. Content Freshness. Another difficult issue with caching transient content is maintaining content freshness, which may result in caching and delivering expired content if not properly considered. For instance, a time-sensitive application such as e-Health would require the most recent content because rapid changes in a patient's health require immediate attention. Similarly, in the event of a smart grid device failure, it is expected that the most up-to-date information will already be stored locally rather than needing to be fetched from the server. Remaining freshness can be calculated using the content lifetime, generation time, and arrival time. For an incoming content, it can be computed as follows:

$$\text{Freshness}_{d(i)} = 1 - \frac{A_i - G_i}{L_i} : \{\text{where}\}\, A_i - G_i < L_i. \quad (1)$$

Equation (1) exhibits the freshness of content d_i at the content router where G_i is content generation time at the producer, A_i is arrival time at caching node, and L_i is the lifetime of the content. If the time difference between A_i and G_i is smaller than L_i, it signifies that the content is still valid and cacheable; otherwise, the content router will discard it. The remaining freshness of the content when it is received by the content router for caching is shown in Figure 4. The producer node in this illustration regularly generates the content every 5 s. Since we assume a link latency of 0.5 s, the same at node N_2 still has 4 s before it expires. The lifetime of the content expires and becomes invalid after 4 s, assuming no new updates are received. Therefore, this content must be removed from the cache, as it should not be used to satisfy future requests.

Cache placement and replacement have received little attention in the literature, and there are very few metrics that aim to strike a balance between cache size, freshness, and the relative popularity of cached and new content.

3.3. PoSiF Caching and Replacement Scheme. The proposed popularity, size, and freshness-based (PoSiF) transient content

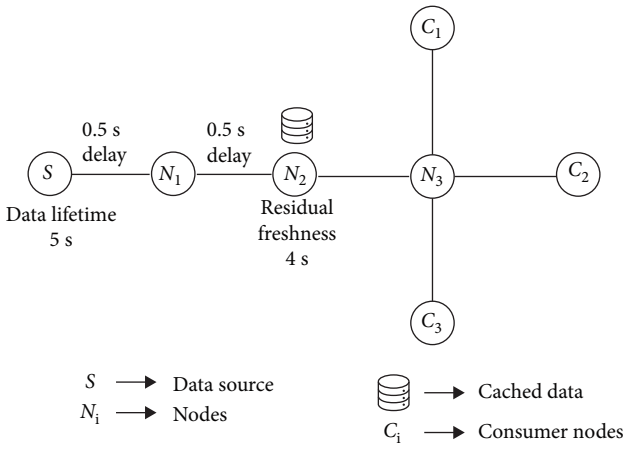

FIGURE 4: Remaining freshness of cached contents.

caching and cache replacement scheme seeks to achieve a balance between cache size, content size, and popularity. The central concept of our proposed scheme is that a tradeoff between content size and content popularity should be considered within the cache in order to achieve better cache hits, reduced node energy consumption, a shorter round trip to the source, and limited bandwidth utilization. Furthermore, IoT devices can benefit greatly from the tradeoff between content size and content popularity.

Table 2 shows the list of variables and a description of these variables used for the purpose of design. Let us define the weight of incoming requests as W_{in} based on their request rate and the size of the content, and can be derived as shown in Equation (2).

$$W_{\text{in}} = P_{\text{in}} * Q_{\text{in}}. \tag{2}$$

On a content router, W_{in} represents the weight assigned to each incoming content. The value of W_{in} should increase when the content has a higher request rate or requires less cache space and decrease otherwise. In Equation (3), P_{in} shows the popularity weight of incoming requests on the caching node, where incoming content with higher popularity may achieve a greater value of P_{in}.

$$P_{\text{in}} = \frac{R_{\text{in}}}{R_{\text{in}} + \sum_{t=1}^{n} [R_t]}. \tag{3}$$

In Equation (4), the variable Q_{in} represents the weight associated with the size of incoming requests on the caching node, which returns a greater weight for smaller incoming content. Here, d_{in} represents the size of incoming content arriving at the caching node, and D_t represents the content size of the cached items.

$$Q_{\text{in}} = 1 - \frac{d_{\text{in}}}{d_{\text{in}} + \sum_{t=1}^{n} [D_t]}. \tag{4}$$

On a caching node the weight of to be evicted cached content (W_{eC}) is determined by Equation (5) to accommodate incoming content. Here, W_{eC} has a greater value when cached content has a greater request rate and uses less cache space on the caching node.

$$W_{eC} = P_{eC} * Q_{eC}. \tag{5}$$

The cumulative P_{eC} and Q_{eC} values for multiple contents, which are present in the content store at present, can be calculated as follows given in Equations (6) and (7), respectively.

$$P_{eC} = \frac{\sum_{i=1}^{k} R_{eC_i}}{R_{\text{in}} + \sum_{t=1}^{n} [R_t]}, \tag{6}$$

$$Q_{eC} = 1 - \frac{\sum_{i=1}^{k} d_{eC_i}}{d_{\text{in}} + \sum_{t=1}^{n} [D_t]}. \tag{7}$$

The popularity weight of cached content that has to be removed is represented by P_{eC} in Equation (6). The request rate of the content that needs to be removed from the content router cache is represented by R_{eC_i}. Furthermore, n denotes the total number of cached items, while k denotes the number of evicted items. A less popular piece of content yields a lower P_{eC} value, and vice versa. In order to accommodate incoming content, the weight of the cached content size that should be evicted is represented by Q_{eC} in Equation (7). Higher cached content occupancy results in lower Q_{eC} values, where d_{eC_i} is the content size that needs to be evicted.

The amount of available cache space at the caching node determines which caching decision is made: directly caching incoming content when sufficient space is available, or replacing existing content with cache replacement methods. As depicted in Figure 5, upon receiving a content packet, a content router will examine cache occupancy and available space. If there is available space in the cache memory of the content router, the packet will be placed there; if not, existing content must be removed from the cache memory. Because different publishers provide a variety of content, the packet size may vary based on the type of data being transmitted. When incoming content is added to the cache, the total size of all cached content cannot exceed the cache limit (CL) defined in Equation (8).

$$D_t + d_{\text{in}} > CL. \tag{8}$$

The CL is the maximum amount of content that a content router can store in its cache. In order to determine

Table 2: Summary of the design parameters.

Variable name	Description
A_i	Arrival time of the generated content at caching node
d_{eC}	Size of the content that need to evict
d_{in}	Size of content arriving at content router
D_t	Size of the item present in content store
G_i	Generation time of the content at producer node
L_i	Lifetime of the content for which it is valid
k	Represents the number of evicted content
n	Represents the total number of cached content
P_{eC}	Weight of the popularity of cached content that need to evict to accommodate incoming content
P_{in}	Weight of the popularity of incoming content
Q_{eC}	Weight of the content sizes of cached content that need to evict to accommodate incoming content
Q_{in}	Weight of the content size of incoming content
R_{eC}	Request rate of the content to evict from content router cache
R_{in}	Request rate of incoming content
R_t	Request rate of the item that present in the content store
t	It denotes the item of the content router cache
W_{in}	Weight of the incoming content
W_{eC}	Weight of the cached content that need to evict to accommodate incoming content

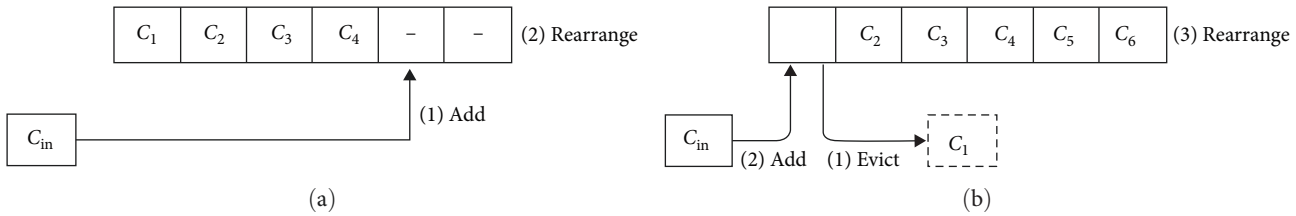

Figure 5: Caching incoming content (a) content store vacant and (b) content store full.

whether the cache has sufficient space to store the incoming content, the content router will initially extract the content size and request rate of the incoming content. If the Expression 8 returns false, it adds the content to the content store without performing an evaluation. Otherwise, the value of W_{eC} is determined in order to decide whether or not to evict specific content from the caching node. When W_{in} is greater than W_{eC}, incoming content will be cached by evicting entries for which the cumulative value W_{eC} was determined to be less.

The content storage is sorted in ascending order of W_{in} whenever content packets are cached or replaced there. Initial estimates assume that both the cumulative request rate R_{eC} and the cumulative content size d_{eC} are zero. It is calculated at each iteration using the Equations (9) and (10). These equations determine the content size that needs to be removed to make place for incoming content as well as the request rate. We calculate the size of the content packet to be removed from the cache to make room for the incoming content packets until sufficient space for the incoming content is discovered by searching for conditions where d_{eC} is smaller than d_{in}.

$$R_{eC} = R_{eC} + R\text{rate}\left(C_{\text{temp}}\right), \tag{9}$$

$$d_{eC} = d_{eC} + \text{size}\left(C_{\text{temp}}\right). \tag{10}$$

Each time an incoming request packet is received, the rate of incoming requests is updated. To accomplish this, Vural et al. [32] suggest updating and maintaining a sliding window of request receipt times. It records the time of the most recent instances of W request packet reception in order to estimate R_{in}. Each caching node keeps a sliding window of this size W that logs the timestamps of all the request packets for each content. As proposed by the authors, the average request rate can be calculated through Equation (11).

$$R_{\text{in}} = N_{\text{req}}/(t_w - t_1), \tag{11}$$

where t_w and t_1 are the last and first timestamps of the incoming request window W, respectively, and N_{req} denotes the number of content requests in the time frame.

3.4. Algorithm. As illustrated in Algorithm 1, intermediate nodes in our proposed PoSiF scheme perform caching and replacement. Before a caching decision is made, incoming content is parsed for R_{in} and d_{in}, as shown in Lines 1–2. The algorithm calculates cache size and limit to determine cache occupancy and total cache size, as shown in Line 3, in order

```
1: function ADD(C_in)
2:   R_in = RRate(C_in)
3:   d_in = Size(C_in)
4:   if Size(Cache) + d_in > CL then
5:     d_eC = 0
6:     R_eC = 0
7:     i = 0
8:     while d_eC < d_in  i < Size(Cache) do
9:       C_temp = Cache[i]
10:      R_eC = R_eC + RRate(C_temp)
11:      d_eC = d_eC + Size(C_temp)
12:      i = i + 1
13:    end while
14:    P_eC = R_eC / (R_in + R_t)
15:    Q_eC = 1 − d_eC / (d_in + D_t)
16:    W_eC = P_eC * Q_eC
17:    P_in = R_in / (R_in + R_t)
18:    Q_in = 1 − d_in / (d_in + D_t)
19:    W_in = P_in * Q_in
20:    if W_in > W_eC then
21:      Add(Cache, C_in)
22:      for j = 0 to i do
23:        C_temp = Cache[j]
24:        Evict(Cache, C_temp)
25:      end for
26:      return true
27:    else
28:      return false
29:    end if
30:  else
31:    Add(Cache, C_in)
32:    return true
33:  end if
34: end function
```

ALGORITHM 1: PoSiF scheme.

to determine whether the cache has sufficient space to store incoming content. If sufficient space is available, incoming content will be cached; otherwise, cached content will be replaced, as shown in Lines 4 through 12. The algorithm iterates through the cache and calculates the weight of cached content P_{eC}, Q_{eC}, and W_{eC} using d_{eC} and R_{eC} as indicated in Lines 13–15. The weight of incoming content, W_{in}, is calculated with P_{in} and Q_{in} as stated in Lines 16–18. If $W_{\text{in}} > W_{eC}$, then new content must be added to the cache; otherwise, as shown in Lines 19–24, no replacement will occur.

3.5. Use Cases with Examples. There are two possible outcomes with the proposed PoSiF approach: either there is no possibility of replacement, or the cached content can be updated to include incoming content. Each of these scenarios is illustrated here.

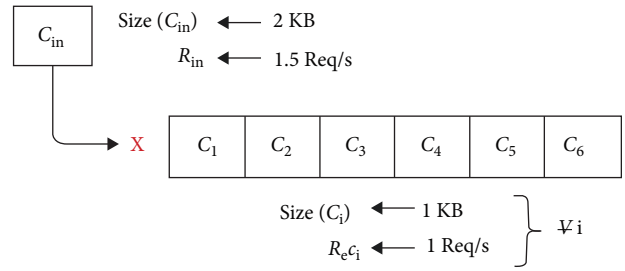

FIGURE 6: Example: no content replacement in cache.

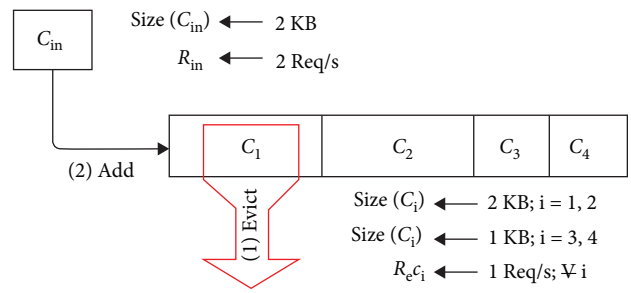

FIGURE 7: Example: content replacement in cache.

3.5.1. No Replacement. In the following two cases, as also illustrated in Figure 5, there will be no replacement: either the content store or cache has enough space to accommodate incoming content, or caching incoming content is no more beneficial than keeping previously cached content. In the scenario depicted in Figure 6, newly arriving content does not take precedence over previously cached content. Existing content will be retained in the cache in this case. Consider a cache with a CL of 6 kB that is completely filled with 1 kB of content. The request rate for each content is 1 request/ms, while the incoming content's d_{in} size is 2 kB, and its R_{in} request rate is 1.5 requests/ms. Therefore, the total request rate R_t and total cached content size D_t are both 6. According to Equation (2), the calculated weight of incoming content W_{in} is 0.15.

In the increasing order of W_{in} of the current contents, our PoSiF scheme sorts and iterates the content store. Since content with a lower weight is at the top, C_1 and C_2 must be evicted from the cache to make room for new content C_{in}, which is of size 2kB; therefore, their cumulative W_{eC} is computed. There are two factors at play here: R_{eC}, which represents the sum of the request rates for both requests and d_{eC}, which represents the sum of the size of the content that will be evicted. Based on the expression 5, the value of W_eC is computed to be 0.20. Thus, it can be seen that the condition $W_{\text{in}} < W_{eC}$ is satisfied. Therefore, incoming content will not be cached, ensuring that the cache satisfies more requests.

3.5.2. With Replacement. With our proposed scheme, existing contents in the cache will be replaced if there is insufficient space to store incoming content. However, it is more beneficial if the cache contains out-of-date information that should be removed. Figure 7 illustrates a scenario in which the cache size is 6 kB and the incoming data is 2 kB. The

Table 3: Simulation parameters.

Parameter name	Description
Publisher	5
Subscriber	45
Cache size	2,048 bytes
Content size	1,024 and 2,048 bytes
Mobility model	Random-direction
Popuparity model	Based on time window used
Connectivity	Wifi
No. of simulation runs	50
Simulation time	100 s

cache is populated with 2 and 1 kB contents whose request rates are 1 request/ms, while the request rate for incoming content is 2 requests/ms. Consider that R_t equals to 4 and D_t equals to 6. Through Equation (2), W_{in} can be calculated to be 0.247. Since the content store is sorted in ascending W_{in} order, the least weighted content is always at the top. In order to accommodate incoming content, C_1 must be removed; consequently, their cumulative W_{eC} is computed, such that d_{eC} is 2, and R_{eC} is 1. Based on the Expression 5, W_{eC} equals 0.124. Clearly, W_{in} is greater than W_{eC}. As a result, incoming content will be cached, indicating that our scheme prioritizes incoming content with a higher weight. This decision enhances future request fulfillment and ensures the caching of content with a small file size and a high request rate. Thus, the proposed PoSiF scheme chooses content that will generate a greater number of cache hits than less popular content of the same size, in this case, C_1.

4. Experimental Results

This section first provides an overview of the experimental setups and existing schemes, including Caching Everything Everywhere (CEE) [17], LCD [25], and CCS/CES [14]. It then compares the proposed scheme alongside these state-of-the-art schemes. Additionally, we compare PoSiF with least recently used (LRU) [44], least frequently used (LFU) [44], and CCS/CES for cache replacement.

4.1. Experimental Setup. The experimental environment is set up for implementing the proposed, and other existing schemes in the ndnSIM [42], a library of NS-3 simulator [43] installed on a machine running Ubuntu 16.04, using the default content store and the Network Forwarding Daemon (NFD). The simulation scenario considered here is similar to the one depicted in Figure 1; however, we randomly deploy 100 nodes over an area of 100×100 m^2, of which 5 are content publishers, and 45 are content subscribers. We assume that the nodes transmit the content of sizes 1,024 and 2,048 bytes at different rates. Various subscribers transmit requests, which are then satisfied by either a caching node or the content publisher. Due to the IoT device's constrained memory, each node's cache size is restricted to 2,048 bytes. The remaining parameter configurations are described in Table 3.

4.2. Existing Schemes. CEE [17] is a caching strategy in ICN networks intended to reduce downstream latency and upstream bandwidth usage. In the CEE scheme, every content router along the path caches a copy of the content. CEE scheme suffers from the problem of caching redundancy. The same request can be answered by multiple nodes simultaneously, and numerous neighboring nodes are expected to get the same data packet. Therefore, the benefit of caching is offset by energy and network resource waste.

The authors in [25] proposedLCD to enhance the content caching structure by minimizing data redundancy. LCD scheme stores a copy of the data beneath the node at which a hit occurs. LCD tends to store content near the producer, and it reduces the producer's workload by caching popular content closer to consumers with each cache hit. However, it increases the number of identical data replications on the return path.

The CCS and CES [14] highlight the distinct requirements of the core and the edge of the network. Equation (12) shows the caching probability of packet d_k, where F_{d_k} is the freshness of packet d_k, I_{d_k} is the number of interest packets in time frame T, P_{th} is popularity threshold, and F_{th} is freshness threshold [14].

$$p_{c,d_k}(\text{CCS}) = \begin{cases} 1, & \text{if} I_{d_k} \geq P_{\text{th}} \;\; F_{d_k} \geq F_{\text{th}} \\ \dfrac{F_{d_k}}{F_{\text{th}}}, & \text{if} I_{d_k} \geq P_{\text{th}} \;\; F_{d_k} \leq F_{\text{th}} \\ 0, & \text{otherwise} \end{cases}. \tag{12}$$

The caching probability is always set to 1 for frequently used, long-lived data packets. Data packets that are not in high demand are never cached, so their caching probability is always 0.

If d_k is the least requested content, the likelihood of caching in the CES scheme is less than 1, and if d_k is the most requested content, the probability of caching is set to 1 in the CES scheme, which is determined by Equation (13).

$$p_{c,d_k}(\text{CES}) = \frac{I_{d_k}}{\max \forall d_i \left(I_{d_i}\right)}. \tag{13}$$

The CCS/CES scheme takes into account the parameters of content freshness and popularity, and caches the content only if it exceeds a predetermined threshold limit in terms of both freshness and popularity. These observations are based on the threshold limits and may not always be suitable. Moreover, they do not consider the content size parameter, leading to inefficient use of cache resources.

4.3. Results and Discussions. Comprehensive simulations were performed to assess the performance of PoSiF and other existing schemes. The experimental outcomes were evaluated using metrics such as cache hit ratio and hop ratio, which were measured under the different request rates. While the cache hit ratio represents the overall performance of the cache and its utilization index, the hop ratio illustrates the overall reduction in path length between the content

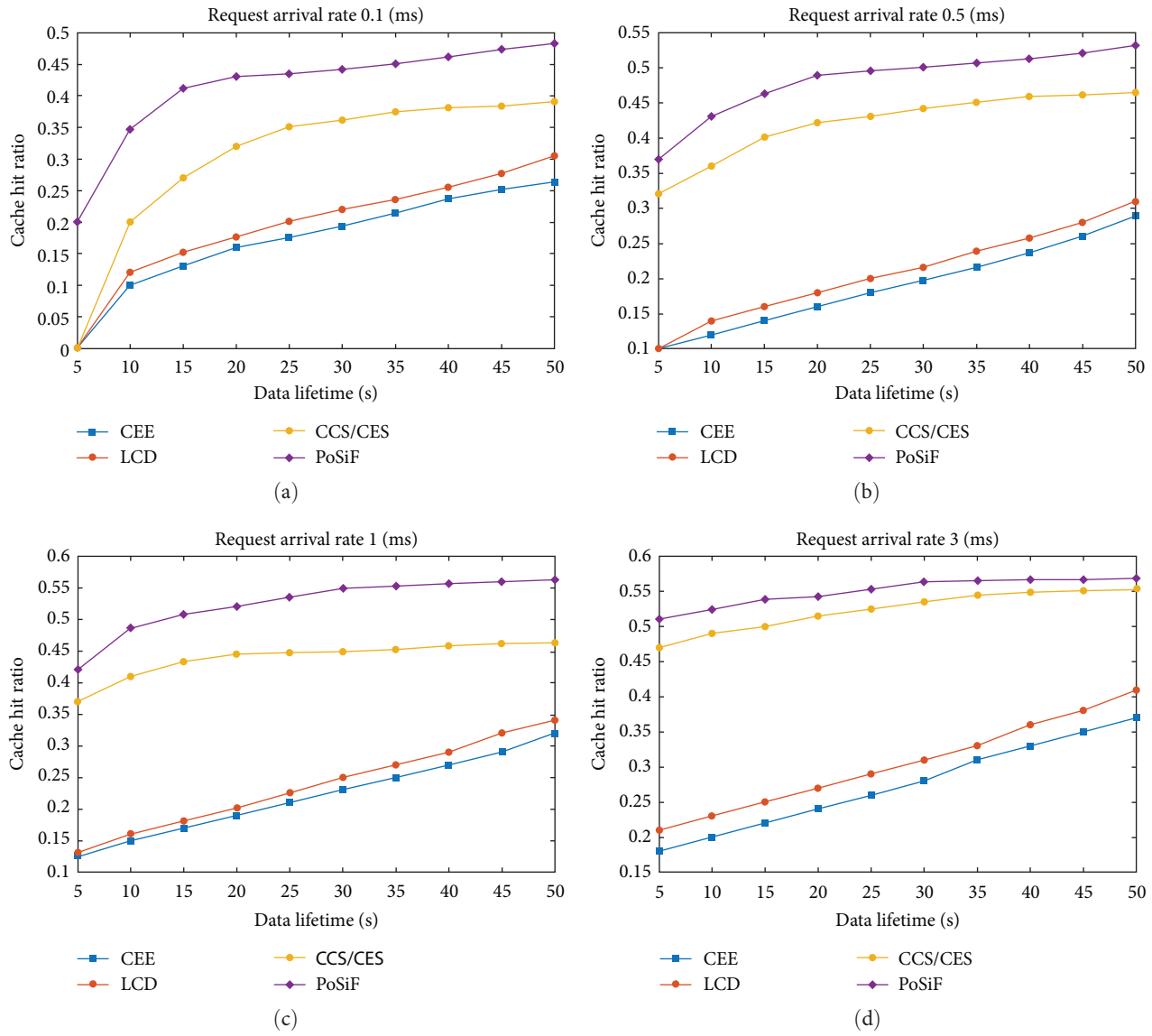

FIGURE 8: Cache hit ratio for request rate (a) 0.1, (b) 0.5, (c) 1, and (d) 3 request/ms.

producer and consumer. Additionally, the average freshness and average distance were calculated for multiple content lifetimes with varying values.

4.3.1. Cache Hit Ratio. The efficacy of a cache system can be evaluated using a metric known as the cache hit ratio. As shown in Equation (14), the cache hit ratio is the proportion of successful cache accesses relative to total cache accesses (hits plus misses).

$$\text{Cache Hit Ratio} = \frac{\text{Cache Hits}}{\text{Total Cache Accesses}}. \quad (14)$$

The cache hit ratio for a request rate of 0.1 request/ms is shown in Figure 8(a). The figure illustrates that CEE and LCD schemes have fewer cache hits than the CCS/CES scheme and the proposed PoSiF scheme. The CEE scheme is the fundamental caching scheme in the ICN network, where a copy of the content is cached at every intermediate node. However, this scheme suffers from content duplication. The LCD method maintains the cache at the network's edge, moving the cache in that direction on each cache hit. The CEE and LCD schemes exhibit a notably lower cache hit ratio in comparison to other schemes as they do not consider content freshness and popularity. Consequently, these schemes may result in the delivery of outdated or caching unpopular content.

CCS/CES scheme, on the other hand, considers the popularity and freshness parameters of the content while making the caching decision, thereby outperforming CEE and LCD schemes. This is due to the fact that caching popular and fresh content can satisfy more requests from the content store. However, the CCS/CES scheme performs worse than PoSiF as they consider content size and popularity only as thresholds while caching, failing to prioritize them so as to maximize cache hits. Therefore, smaller contents may be

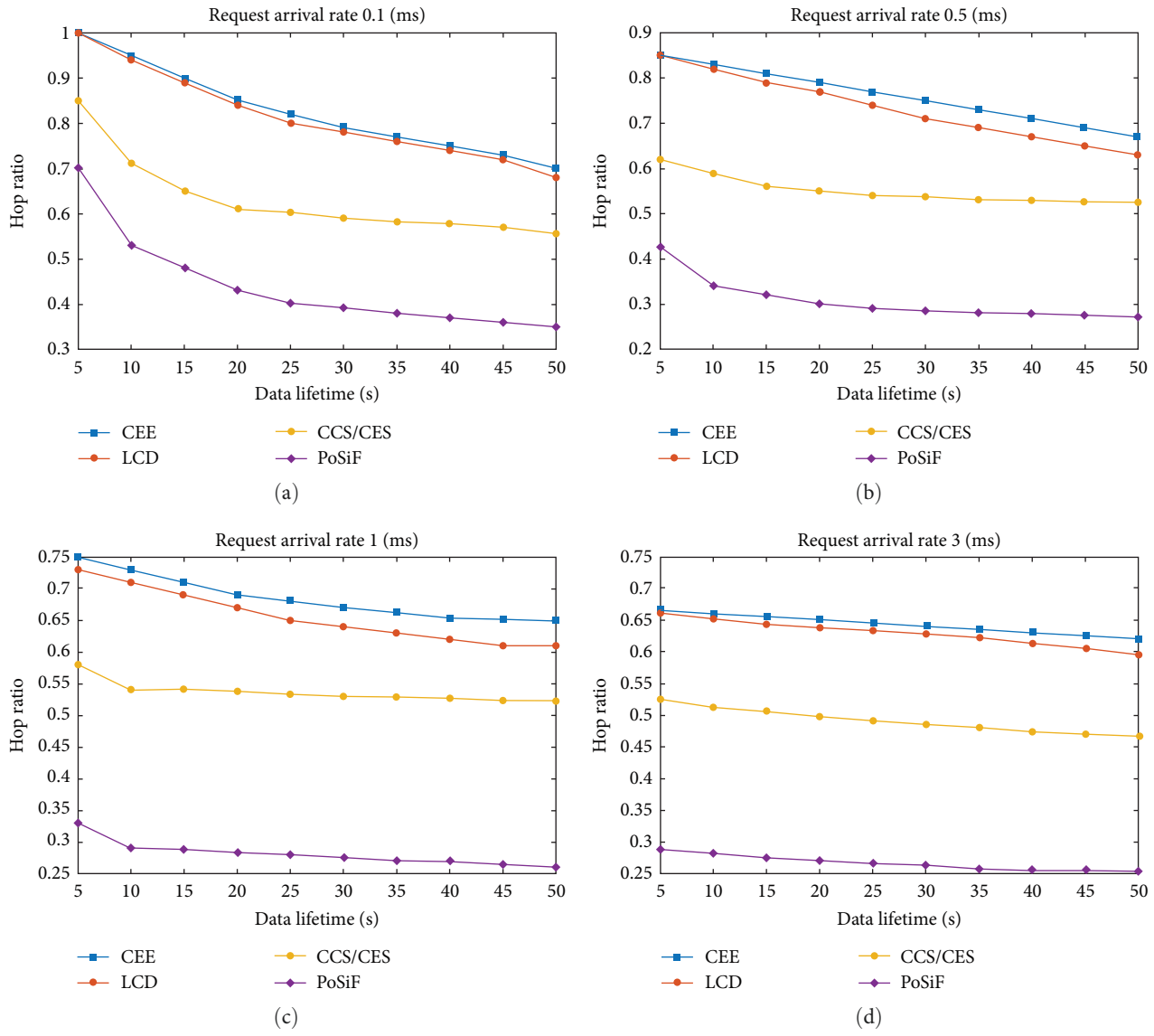

Figure 9: Hop ratio for request rate (a) 0.1, (b) 0.5, (c) 1, and (d) 3 request/ms.

evicted due to the insertion of very large content with marginally higher content popularity, thereby affecting the cache hit ratio negatively. In contrast, the weight of incoming content is computed based on its popularity, freshness, and content size in PoSiF, prioritizing smaller and more popular content over the others while caching. It can be observed from Figure 8(a) that the PoSiF scheme is approximately 40% more efficient than CCS/CES.

Similar trends can be observed for request rates of 0.5, 1.0, and 3.0, as depicted in Figure 8(b)–8(d), where the improvement for the proposed scheme over CCS/CES schemes is approximately 16.4%, 23.0%, and 7.0%, respectively, attaining an overall improvement of 22% in terms of cache hits. It can further be observed that the cache hit ratio increases with the increase in content lifetime and popularity. As the lifetime increases, a greater number of requests are satisfied by the caches over a longer period of time. Because cache size is fixed, the cache hit ratio may not increase significantly with an increase in content lifetime, and similar cached content may not differ significantly in terms of cache hits.

4.3.2. Hop Ratio. The hop ratio is defined as the ratio between the average hop count and the physical distance between the producer and the consumer. It attempts to demonstrate, on average, how close the contents are per unit distance between the source and the destination as a result of caching. The comparative results between CEE, LCD, CCS/CES, and PoSiF for a request rate of 0.1 request/ms are shown in Figure 9(a). The figure illustrates that, while the CEE and LCD schemes have a higher hop ratio than CCS/CES and PoSiF, the PoSiF scheme shows approximately a 30% reduction compared to the current CCS/CES scheme. This is due to the fact that our proposed PoSiF scheme prioritizes small, popular content to remain in the cache for longer. CCS/CES, on the other hand, do not consider content size metrics and

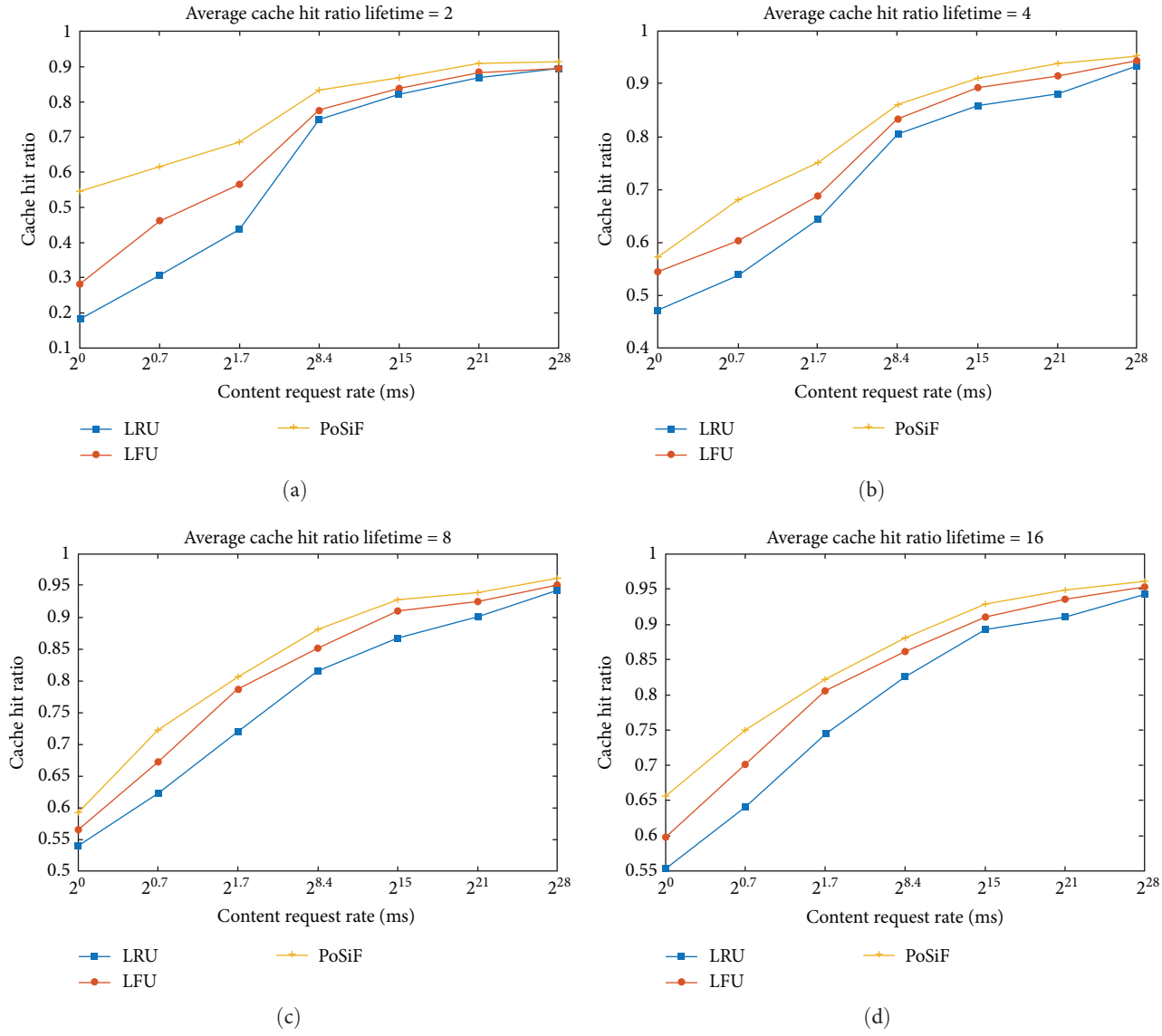

Figure 10: Cache hit ratio for lifetime (a) 2, (b) 4, (c) 8, and (d) 16 s.

place a greater emphasis on freshness and popularity metrics, causing some content routers to drop small content with similar or slightly lower popularity. This leads to accommodating a smaller number of items in the cache, thereby reducing the chance of cache hits at nearby nodes.

Similar trends can also be observed for the request rates of 0.5, 1.0, and 3.0, as depicted in Figure 9(b)–9(d), where the CEE and LCD schemes attain less hop ratio than the CCS/CES scheme, while improvement for PoSiF is about 44%, 48%, and 48%, respectively, compared to the CCS/CES scheme. Therefore, it can be concluded from the comparisons of the proposed PoSiF scheme and the existing CCS/CES scheme that the former achieves an average of 40% lower hop ratio. It is to be noted that Figure 9 shows a higher hop ratio for a low-content lifetime, where the hop ratio decreases with the increase in content lifetime and popularity. In these cases, most of the in-network cached copies of the content expire before they can satisfy another request if both the lifetime and the request rate are very short. As a result, the contents are delivered directly from the producer, which increases the hop ratio.

4.4. Efficacy of Cache Replacement. Several challenges are faced while tackling content replacement, including maintaining cache consistency, evictions, and the content's freshness. IoT devices may have limited storage capacity for caching, buffering, or storing data temporarily. Large content sizes can occupy a large portion of available storage space, limiting the device's capacity to locally store and interpret data. The popularity of content, however, has a direct effect on the efficiency of caching solutions. Delivering frequently requested content from local caches reduces the need to retrieve data from remote sources, thereby decreasing latency. To improve cache utilization, less popular content should be replaced with highly popular content. In instances where IoT content is transient, its freshness must be

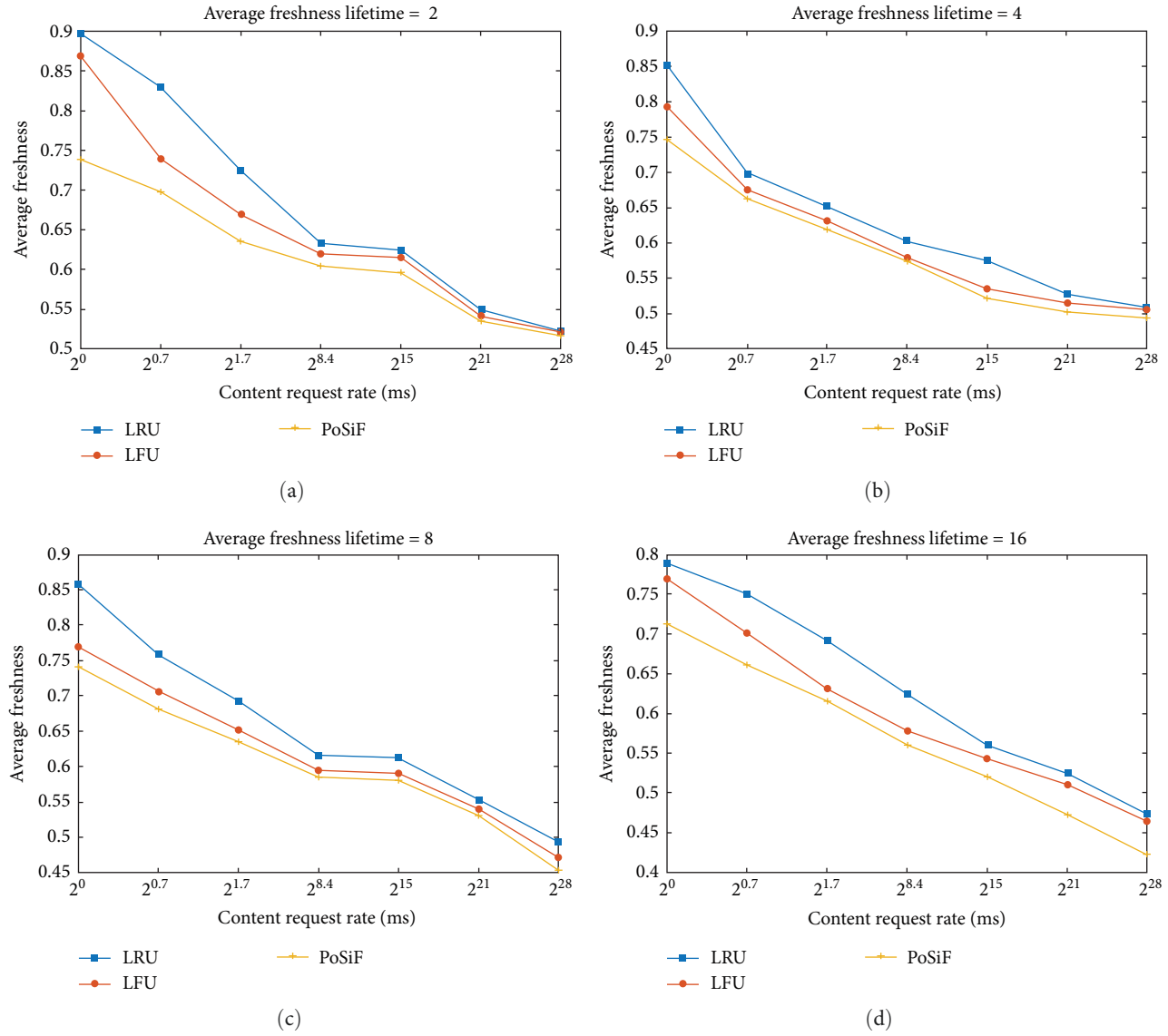

Figure 11: Retrieved freshness for lifetime (a) 2, (b) 4, (c) 8, and (d) 16 s.

considered when making caching decisions. It indicates the amount of time since the data were generated, updated, or received. Transient content is only valid for a limited time, after which it should be removed or replaced with new content. IoT devices have limited cache memory; consequently, smaller content with greater popularity and longer remaining freshness will be prioritized in the cache. Therefore, maintaining the relationship between these three parameters becomes the fundamental criterion.

Cache replacement plays a significant role when considering the caching of transient content. LRU [44] and LFU [44] are two of the most prevalent cache replacement policies. LFU is more efficient than LRU and can circumvent the issue where periodic or accidental operations cause the cache hit rate to decrease [44]. To assess the efficacy of our proposed scheme, we compare the PoSiF scheme with both LRU and LFU cache replacement schemes in terms of the cache hit rate, cache freshness, and average distance, as depicted in Figures 10(a) through 11(d). In order to accommodate a wide range, the values for the incoming request rates are shown in the log scale in these figures, while the value of x is assumed to be 0.03 requests/ms.

4.4.1. Cache Hit Ratio. First, we compare the cache hit ratio of LRU and LFU with our proposed scheme for transient content with various lifetimes of 2, 4, 8, and 16 s. Each node determines the request rates of cached contents and computes the average request rate of cached contents. In Figure 10(a), a comparison of the cache hit ratio between the three schemes is depicted for a cache lifetime of 2 s. The figure illustrates that the LFU performs about 12% better than the LRU cache replacement scheme. On the other hand, the PoSiF scheme performs approximately 33% and 19% better than the LRU and LFU schemes, respectively. This is because our proposed PoSiF scheme prioritizes higher weight content inside the cache and evicts the small-weight

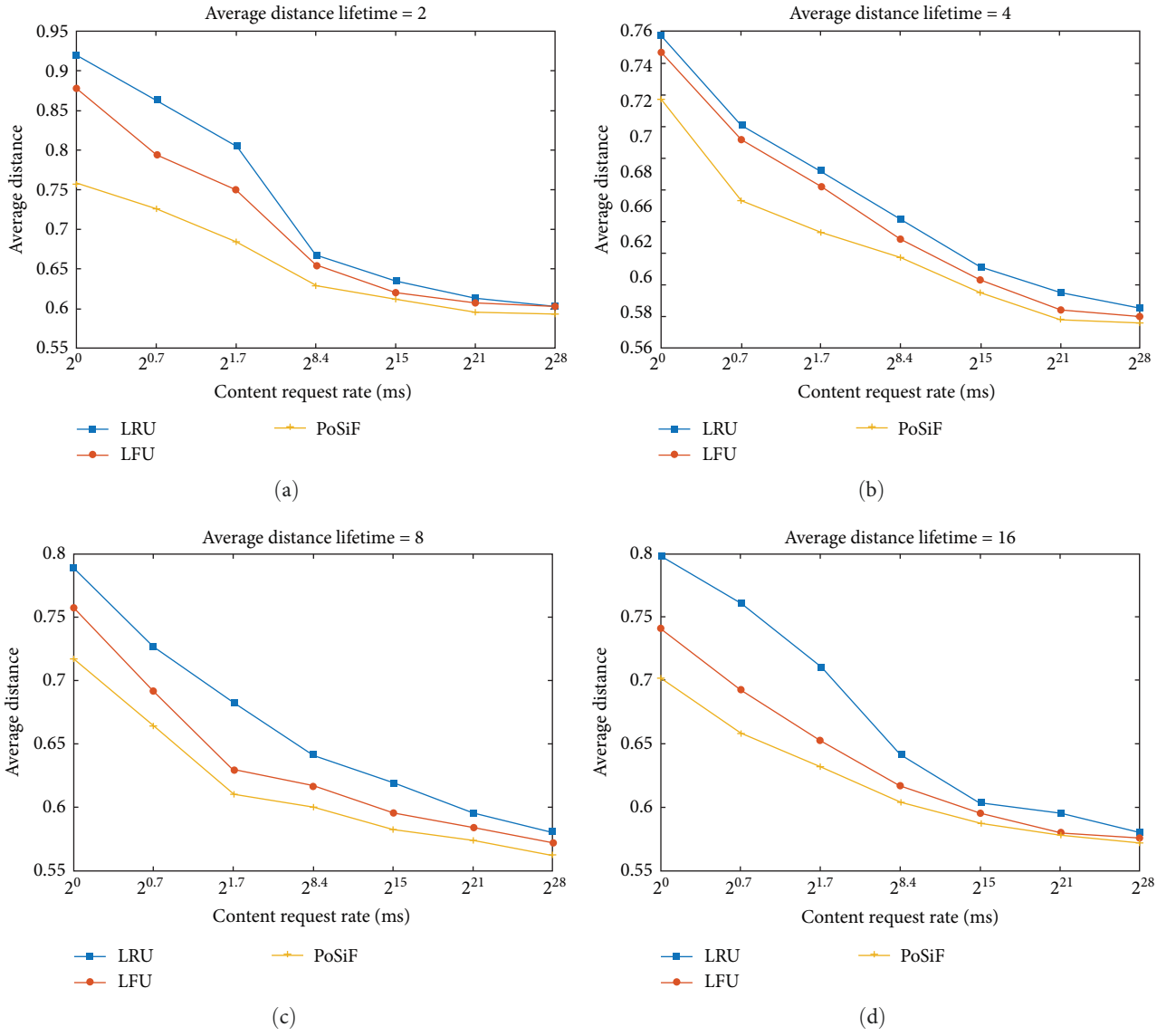

FIGURE 12: Average hop distance for lifetime (a) 2, (b) 4, (c) 8, and (d) 16 s.

contents. On the other hand, LRU and LFU perform well in some applications but do not adapt effectively to changing access patterns.

Figure 10(b)–10(d) shows analogous observations for lifetimes 4, 8, and 16 s, respectively. Here, the approximate average improvements by the proposed scheme compared to LRU and LFU schemes are about 13% and 5%, 8% and 3%, 17%, and 3%. Based on the comparisons between the proposed PoSiF scheme and the existing LRU and LFU schemes, it can be concluded that the proposed scheme achieves an average of about 18% and 7.2% more cache hits than LRU and LFU, respectively. In addition, it is evident from all four sets of results that the cache hit ratio for all three strategies converges as the request rate rises, as the majority of requests is directed toward the few most popular contents, which are more likely to be cached under all three systems.

4.4.2. Retrieved Freshness. Retrieved freshness represents the content's remaining lifetime as it arrives at an intermediate node, and therefore, the content's freshness is greatest at its source and decreases as it propagates to more distant nodes. The retrieved freshness can be calculated with the help of Equation (1). The comparison of the retrieved freshness between the proposed method, the LRU, and the LFU schemes for a content lifetime of 2 s is shown in Figure 11(a). The figure illustrates that the LFU cache replacement scheme gives 4.76% lower freshness than the LRU scheme. On the other hand, the PoSiF scheme attains 11.60% and 7.18% lower retrieved freshness than the LRU and LFU schemes, respectively. Similar patterns can be seen for lifetimes of 4, 8, and 16 s, as shown in Figure 11(b)–11(d), respectively, where the proposed scheme improves on LRU and LFU schemes by 4.41% and 0.8%, 5.77% and 0.86%, 10.72%, and 3.98%,

respectively. Based on the comparisons between the proposed PoSiF scheme and the existing LRU and LFU schemes, it can be concluded that the former achieves, on average, 8.12% less freshness than the LRU scheme and 3.2% less than the LFU scheme.

As the PoSiF aims to satisfy more requests from cache-based content stores than the other schemes, it tends to deliver valid but relatively older content which negatively impacts the retrieved freshness of the delivered content. As the content retention periods increase, the PoSiF scheme satisfies more requests from cache-based content stores than the other schemes, decreasing the freshness of the content retrieved. On the other hand, LRU and LFU schemes prioritize regularly requested and frequently accessed contents, respectively. This leads to discarding contents from the cache irrespective of their remaining lifetime or their size, thereby increasing the retrieved freshness. The results also demonstrate that a smaller lifetime and lower request rates result in a greater degree of retrieval freshness. If both the lifetime and request rates are relatively low, the majority of in-network cached copies of the content will expire prior to serving another request. This results in most content being retrieved directly from the producer and arriving at the consumer with greater freshness. In contrast, as the packet's lifetime and request rate increase, the freshness of the retrieved content decreases, indicating that a larger proportion of content requests will be served from the cache.

4.4.3. Average Distance. The hop distance between the content requester and content producer is deemed the maximum distance, where the hop count is decreased when a request is fulfilled by an intermediate cache node. The average distance for a lifetime of 2 s is shown in Figure 12(a). The figure illustrates that the LFU cache replacement scheme gives a 2.4% lower hop distance than the LRU scheme. On the other hand, the PoSiF scheme gives 10.7% and 8.4% lower average distances than the LRU and LFU schemes, respectively. This is due to the fact that popular requests are satisfied from the intermediate caches, resulting in fewer hops in PoSiF. It can also be observed that the distance decreases with the increase of the content lifetime for all three schemes. This happens because, within the large time frame, more requests will get satisfied from the cache.

Figure 12(b)–12(d) depict similar trends for lifetimes 4, 8, and 16, where the improvement for the proposed PoSiF scheme compared to the LRU and LFU schemes is approximately 4.7% and 3.2%, 7.1% and 1.8%, and 8.1% and 2.8%, respectively. Therefore, it can be concluded that the proposed PoSiF scheme reduces hops by an average of 7.6% when compared to the LRU scheme and 4.1% when compared to the LFU scheme.

5. Conclusions and Future Works

Although it is clear that considering cache size and freshness when caching transient content in IoT is useful for various reasons, the great majority of prior efforts hase not considered essential parameters as design requirements. Furthermore, by making the content size, the primary consideration in caching decisions and simultaneously taking into account the request rates or the popularity of the content, the issue of low-cache hits and large hop counts can be somewhat mitigated. This article offers a transient content caching and cache replacement method (PoSiF) based on popularity, size, and freshness for ICN-enabled IoT devices in order to accomplish these objectives. The experimental results demonstrate significant improvements in hop counts, cache hits, freshness, and distance over other existing and state-of-the-art schemes.

Transient content caching is a valuable tool for applications that require frequent access to the updated data, such as stock market tickers, sports scores, and weather predictions. The primary objective of this technique was to efficiently utilize the content store. However, it may not yield satisfactory outcomes when implemented in scenarios where content sizes are uniformly distributed over the network. Furthermore, it is worth noting that there are prospects for further study aimed at enhancing the functionality of IoT devices with limited capabilities. Future research should prioritize investigating several elements, including energy efficiency considerations, effective management of large interest tables, and robust security measures. These areas have the potential to significantly enhance the usability and functionality of IoT technology. Adapting content delivery to each user's desires, habits, and surroundings is a potential area of research for the near future. This may entail dynamically adjusting cache content based on user interactions or targeting certain user segments with cached content. Security and privacy should be taken into account during transient content caching. Therefore, caching techniques that safely handle sensitive data and prevent data leaks or illegal access may be another focus of the future studies.

Data Availability

Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Cisco, "Annual internet report (2018-2023) white paper," 2018, https://www.cisco.com/, (Accessed: 2020-09-30).

[2] D. Miorandi, S. Sicari, and F. De Pellegrini, "Internet of things: vision, applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[3] B. Panigrahi, S. Shailendra, H. K. Rath, and A. Simha, "Universal caching model and markov-based cache analysis for information centric networks," *Photonic Network Communications*, vol. 30, no. 3, pp. 428–438, 2015.

[4] G. Xylomenos, C. N. Ververidis, V. A. Siris et al., "A survey of information-centric networking research," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.

[5] A. V. Vasilakos, Z. Li, G. Simon, and W. You, "Information centric network: research challenges and opportunities,"

Journal of Network and Computer Applications, vol. 52, pp. 1–10, 2015.

[6] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox, "Information-centric networking: seeing the forest for the trees," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, pp. 1–6, ACM, Cambridge Massachusetts, USA, 2011.

[7] I. Abdullahi, S. Arif, and S. Hassan, "Survey on caching approaches in information centric networking," *Journal of Network and Computer Applications*, vol. 56, pp. 48–59, 2015.

[8] M. Amadeo, C. Campolo, A. Iera, and A. Molinaro, "Named data networking for IoT: an architectural perspective," in *2014 European Conference on Networks and Communications (EuCNC)*, pp. 1–5, IEEE, Bologna, Italy, 2014.

[9] C. Gündoğan, T. C. Schmidt, M. Wählisch, C. Scherb, C. Marxer, and C. Tschudin, "Information-centric networking (ICN) adaptation to low-power wireless personal area networks (LoWPANs)," 2021, https://www.rfc-editor.org/info/rfc9139, (Accessed: 2021-12-10).

[10] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache "less for more" in information-centric networks (extended version)," *Computer Communications*, vol. 36, no. 7, pp. 758–770, 2013.

[11] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian et al., "Less pain, most of the gain: incrementally deployable ICN," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 147–158, 2013.

[12] E. Baccelli, C. Mehlis, O. S. Hahm, C. Thomas, and M. Wählisch, "Information-centric networking in the IoT: experiments with NDN in the wild," in *Proceedings of the 1st International Conference on Information-centric Networking*, pp. 77–86, ACM, Paris, France, 2014.

[13] J. A. Stankovic, "Research directions for the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3–9, 2014.

[14] M. Amadeo, C. Campolo, G. Ruggeri, and A. Molinaro, "Beyond edge caching: freshness and popularity aware IoT data caching via NDN at internet-scale," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 352–364, 2022.

[15] T. Koponen, M. Chawla, B.-G. Chun et al., "A data-oriented (and beyond) network architecture," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 181–192, 2007.

[16] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos, "Developing information networking further: from PSIRP to PURSUIT," in *Broadband Communications, Networks, and Systems. BROADNETS 2010*, vol. 66 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 1–13, Springer, Berlin, Heidelberg, 2014.

[17] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, pp. 1–12, ACM, Rome, Italy, 2009.

[18] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012.

[19] N. Fotiou and G. C. Polyzos, "Realizing the internet of things using information-centric networking," in *10th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, pp. 193-194, IEEE, Rhodes, Greece, 2014.

[20] D. Gupta, S. Rani, S. H. Ahmed, and R. Hussain, "Caching policies in NDN-IoT architecture," in *Integration of WSN and IoT for Smart Cities*, EAI/Springer Innovations in Communication and Computing, pp. 43–64, Springer, Cham, 2020.

[21] O. Waltari and J. Kangasharju, "Content-centric networking in the internet of things," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 73–78, IEEE, Las Vegas, NV, USA, 2016.

[22] Y. Hua, L. Guan, and K. G. Kyriakopoulos, "A fog caching scheme enabled by ICN for IoT environments," *Future Generation Computer Systems*, vol. 111, pp. 82–95, 2020.

[23] M. Amadeo, C. Campolo, J. Quevedo et al., "Information-centric networking for the internet of things: challenges and opportunities," *IEEE Network*, vol. 30, no. 2, pp. 92–100, 2016.

[24] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proceedings of the Second Edition of the ICN workshop on Information-Centric Networking*, pp. 55–60, ACM, Helsinki, Finland, 2012.

[25] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD interconnection of LRU caches and its analysis," *Performance Evaluation*, vol. 63, pp. 609–634, 2006.

[26] M. Zhang, H. Luo, and H. Zhang, "A survey of caching mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1473–1499, 2015.

[27] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "Wave: popularity-based and collaborative in-network caching for content-oriented networks," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 316–321, IEEE, Orlando, FL, USA, 2012.

[28] Z. Ming, M. Xu, and D. Wang, "Age-based cooperative caching in information-centric networking," in *23rd International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–8, IEEE, Shanghai, 2014.

[29] J. Quevedo, D. Corujo, and R. Aguiar, "Consumer-driven information freshness approach for content-centric networking," in *International Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 482–487, IEEE, Toronto, ON, Canada, 2014.

[30] Z. Zhang, H. Ma, and L. Liu, "Cache-aware named-data forwarding in internet of things," in *Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, San Diego, CA, USA, 2015.

[31] M. A. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "Caching in named data networking for the wireless internet of things," in *International Conference on Recent Advances in Internet of Things (RIoT)*, pp. 1–6, IEEE, Singapore, 2015.

[32] S. Vural, N. Wang, P. Navaratnam, and R. Tafazolli, "Caching transient data in internet content routers," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, pp. 1048–1061, 2017.

[33] Z. Zhang, C.-H. Lung, I. Lambadaris, and M. St-Hilaire, "IoT data lifetime-based cooperative caching scheme for ICN-IoT networks," in *International Conference on Communications (ICC)*, pp. 1–7, IEEE, Kansas City, MO, USA, 2018.

[34] H. Zhu, Y. Cao, X. Wei, W. Wang, T. Jiang, and S. Jin, "Caching transient data for internet of things: a deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2074–2083, 2019.

[35] I. Ud Din, S. Hassan, A. Almogren, F. Ayub, and M. Guizani, "PUC: packet update caching for energy efficient IoT-based

information-centric networking," *Future Generation Computer Systems*, vol. 111, pp. 634–643, 2020.

[36] G. F. Marias, N. Fotiou, and G. C. Polyzos, "Efficient information lookup for the internet of things," in *International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1–6, IEEE, San Francisco, CA, USA, 2012.

[37] Y. Song, H. Ma, and L. Liu, "Content-centric internetworking for resource-constrained devices in the internet of things," in *International Conference on Communications (ICC)*, pp. 1742–1747, IEEE, Budapest, Hungary, 2013.

[38] S. Vural, P. Navaratnam, N. Wang, C. Wang, L. Dong, and R. Tafazolli, "In-network caching of internet-of-things data," in *2014 IEEE International Conference on Communications (ICC)*, pp. 3185–3190, IEEE, Sydney, NSW, Australia, 2014.

[39] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, and K. Drira, "How to cache in ICN-based IoT environments," in *14th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1117–1124, IEEE, Hammamet, Tunisia, March, 2017.

[40] S. Fatale, R. S. Prakash, and S. Moharir, "Caching policies for transient data," *IEEE Transactions on Communications*, vol. 68, no. 7, pp. 4411–4422, 2020.

[41] M. Amadeo, G. Ruggeri, C. Campolo, A. Molinaro, and G. Mangiullo, "Caching popular and fresh iot contents at the edge via named data networking," in *IEEE INFOCOM Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 610–615, IEEE, Toronto, ON, Canada, 2020.

[42] ndnSIM, "Simulator named data networking," 2020, https://ndnsim.net/2.4/, (Accessed: 2020-01-02).

[43] NS3, "NS-3 network simulator," 2020, https://www.nsnam.org/, (Accessed: 2020-01-02).

[44] G. Hasslinger, J. Heikkinen, K. Ntougias, F. Hasslinger, and O. Hohlfeld, "Optimum caching versus LRU and LFU: Comparison and combined limited look-ahead strategies," in *16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 1–6, IEEE, Shanghai, China, 2018.

[45] G. Dhawan, A. P. Mazumdar, and Y. K. Meena, "CNCP: a candidate node selection for cache placement in ICN-IoT," in *IEEE 6th International Conference on Information and Communication Technology (CICT)*, pp. 1–6, IEEE, Gwalior, India, 2022.