Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4979057, 12 pages
https://doi.org/10.1155/2022/4979057

Research Article

An Information-Centric Network Caching Method Based on Popularity Rating and Topology Weighting

Yaxin Chang,[1] Jiafei Guo,[2] Hanbo Wang,[2] Dapeng Man,[2] and Jiguang Lv[2]

[1]*China Energy, Beijing 100011, China*
[2]*Information Security Research Center, Harbin Engineering University, Harbin 150001, China*

Correspondence should be addressed to Jiguang Lv; lvjiguang@hrbeu.edu.cn

Received 6 June 2022; Revised 13 July 2022; Accepted 22 July 2022; Published 11 August 2022

Academic Editor: Lisheng Fan

Ubiquitous caching is a feature shared by all proposed information-centric network (ICN) architectures. Prioritising storage resources to popular content in the network is a proven way to guarantee hit rates, reduce the number of hops forwarded, and reduce user request latency. An ideal ICN caching mechanism should make the best use of relevant information such as content information, network state, and user requirements to achieve optimal selection and have the ability to adaptively adjust the decision cache content for dynamic scenarios. Since router nodes have limited cache space, it is then useless to accurately predict the popularity of the content with very low popularity, as this content has no chance of being cached. A more effective approach is to focus on content with high popularity that influences caching decisions. As for different nodes, they have different sets of popular content, and using this property, this paper designs a caching method based on the popularity hierarchy with topological weights. The method considers managing the cached content in nodes with a hierarchy of popularity and improving their distribution in terms of the importance of the nodes' position in the network. Finally, the scheme is simulated by changing the parameter settings under different actual topologies on the simulation platform to confirm the feasibility of the scheme.

1. Introduction

Different from the TCP/IP architecture, ICN does not need to establish a connection between the source address and the destination address before data transmission but directly finds and receives content in the way of user-initiated requests. The router can store the content forwarded through it for a period of time (this storage depends on the size of the cache space and the replacement policy as well as the timeliness of the content) and make it available to the requesting consumer on a hop-by-hop basis. The network's built-in cache is therefore an important feature of ICN networks, and it is one of the two goals of the proposed future Internet. When a router node serves a query for content, the local cache node may have a copy in its content store (henceforth CS) for the purpose that if a new request for that content arrives, it can be satisfied locally, rather than being forwarded to the source-destination server node for network resources [1]. This approach will improve content hit rates, increase bandwidth utilization, and reduce the number of hops of data forwarding and content retrieval latency, which is one of the biggest differences between ICN and TCP/IP architectures other than the protocol stack.

The idea behind caching is that the content being cached is assumed to be potentially accessible in the future, and storing them on the router reduces the average hop count and reduces the load from redundant network traffic. The primary advantage of serving content from the router to the user is that the user experiences lower latency, thanks to the fact that not all requests need to be routed to the content host; the content may be cached on the router during the return of the packet to the user, depending on the cache management policy. Secondly, because routers close to the user (i.e., at the edge of the network) can respond to a large proportion of requests, this alleviates network congestion to a certain extent. However, similar to the

CDN technology in the TCP/IP network, the ICN network cache also faces the problem of low cache utilization caused by the misuse of the cache.

In recent years, the ICN caching technology based on content popularity has been widely proposed to solve the problem of cache misuse by calculating the historical popularity as an indicator for deciding the content to be cached by nodes in the future through a statistical approach to improve the efficiency of cache usage. However, in practice, the impact of the statistical popularity method on the cache performance of router nodes has not been considered, and there is a problem of taking up a lot of computing resources and space and even pulling down the retrieval rate. The caching strategy that uses content popularity alone as a decisive indicator of whether to cache content does not take into account the issue of content redundancy and whether it is adapted to the network topology in which the node is located, as it can only significantly reduce user request latency if the cached content is sunk to an edge node that is closer to the users.

In order to improve the cache utilization and improve the user's network experience as much as possible, ICN needs an effective cache mechanism. In an actual network, different nodes are interested in different contents, so the content preferentially stored by different nodes is also different. According to the above reasons, this paper proposes a caching scheme based on content popularity and topology weighting.

The main contributions are as follows:

(i) A popularity-based cache hierarchy is proposed to divide contents with different popularity into four levels

(ii) It is proposed to combine the popularity level and topology weight as the cache priority index of nodes to make reasonable cache decisions

(iii) Through comparative experiments, the feasibility and effectiveness of the caching mechanism in the ICN environment are verified

2. Related Work

Caching mechanisms commonly used in existing architectures implementing ICN networks include LCE [2], LCD [3], MCD [4], and Prob [3], but these caching methods are generally based on the migration of web caching to ICNs and do not fit perfectly with the characteristics of ICNs. For example, LCE is a "ubiquitous" copy, which improves cache hit rates and reduces content retrieval time but does not provide efficient management of network resources. Content that will not be revisited leaves a large number of copies in the global network, and storing them in cache space is a misuse of cache resources. Although ICN maintains some caching strategies at the beginning of its design, these methods all have certain problems: Prob can be seen as an improvement on the random form of LCE, but nodes can only keep local copies with probability *p*. LCD reduces redundancy by sinking the content cache to the next hop node in the cache hit but still causes all nodes in the communication path to cache the same content, using bandwidth to the maximum. MCD reduces the number of identical copies between the requesting host and the server but increases request latency due to eviction operations.

Content popularity distribution is an important network parameter that affects the performance of caching policies, and caching policy design based on content popularity is becoming a common approach. Existing caching policies based on content popularity are based on the analysis of user preferences, and models are trained by collecting various metrics to distinguish or predict popular content. These methods can increase the cache hit rate of network edge nodes, but there is a problem that training the model requires a lot of data collection and resource-intensive parameter update and also fails to effectively utilize the cache resources of nodes in the upstream network.

Several studies [5–7] use request frequency, content timestamp, content quantity, content name, etc., as popularity evaluation metrics. A piece of content is only cached when its popularity exceeds a set threshold. However, if certain content is heavily accessed during a certain time period, this will lead to an increase in the threshold value, new requested content will not enter the cache, and the cache hit rate will be reduced as a result.

Zhang et al. [5] in the Optimal Cache Placement based on Content Popularity (OCPCP) policy calculate the popularity of incoming content based on the stored content and store new content based on its popularity value. OCPCP makes caching decisions by considering only the content request records on a single node. That is, if the number of content requests is high, it has higher popularity and is therefore considered for caching.

Time-aware least recent use (TLRU) [6] is a content lifecycle-aware eviction policy that improves on the LRU cache management policy, where the timestamp of arriving content is calculated locally by the cache node. If the average request time is less than the timestamp of the stored content, the arriving content is cached. If space is available in the cache, TLRU stores the content; otherwise, it applies a simple LRU to the cached content, creating space for new arrivals.

The fine-grained popularity-based cache (FGPC) [7] will cache all incoming content if the control center of the network node is not full. Otherwise, it stores only popular content and periodically modifies the content popularity threshold. When forwarding content downstream or receiving content from upstream, three statistics are updated in this policy, namely, the content counter, content name, and timestamp. If the value of the new content is greater than the predefined threshold, FGPC uses the LRU policy to cache the new content in the CS; otherwise, it is ignored.

The ProNDN scheme proposed by Pu [8] is a combination of per-network state forwarding and in-network caching policies. Its collaborative data caching consists of two schemes: CacheData and CacheFace. In short, it combines both schemes with the default in-network cache. Popular data is cached using CacheData; if the router holding the data is closer to the edge router than to the producer, then CacheFace is used to cache the interface. Otherwise, the default caching scheme is used.

Zhang and Wang [9] achieve collaborative content and space utilization between local and neighbouring nodes by caching the content replaced by the local node in the neighbouring node's cache. When a local node receives a packet of interest, the node with which it shares information can satisfy that content. Zheng et al. [10] proposed a noncooperative game theoretic-based ICN pricing model for free content to address the problem that existing ICN pricing mechanisms only study paid content and ignore free content in the network. Considering the coexistence of paid and free content in real networks, the authors analyze the impact of caching and pricing on the revenue of all entities and develop a win-win pricing strategy.

Liu and Han [11] focus on allocating cache sizes for each node within a given total cache space budget. The authors explored the impact of heterogeneous cache allocation on content dissemination under the same ICN infrastructure, quantifying the importance of nodes in terms of content dissemination and network topology. They implemented a hierarchical approach based on content dissemination, then developed a set of weight calculations for these hierarchies, and provided cache space allocation per node to assign the total cache space budget to each node in the network. Shekhawat et al. [12] proposed a heterogeneous path cache budget allocation method based on the reference location of nodes to assign caches to content stores. The experimental results were compared with a traditional on-path caching decision (the data is sent back according to the interest forwarding path) mechanism and achieved a 14% improvement in the cache hit rate.

Chiu et al. [13] investigated a two-tier caching scheme where administrative autonomy was achieved by adding nonpath collaborative caches within the management node AS to eliminate redundancy.

Alhowaidi et al. [14] devised a centralized approach to managing/mapping the current content of CS using SDN controllers. SDN controllers are effectively used to analyze the network state and redirect incoming interest to off-path routers that have cached the requested content. This approach enhances the data by allowing NDN consumers and NDN routers to fetch content from multiple off-path locations based on the network state.

While these approaches are clearly layered, real-world networks are more complex than experimental ones, and the way a piece of content is stored during delivery may be a mix of these characteristics. In some mobile network topologies, interconnected communication nodes are constantly changing, and changes to the cached content set become frequent as a result. These situations illustrate the complexity of caching in real networks, increasing the redundancy and replacement of cached copies. Therefore, business-oriented requirements need to be considered when designing caching solutions, as it is difficult to find a universal solution.

3. Analytical Methods

This paper proposes an on-path caching policy algorithm PT-Cache (popularity-topology cache) that uses packet popularity ratings to make them compete directly on caching nodes based on their potential to save forwarding hops for future packets of interest. The strategy not only improves cache hit rates but also looks to reduce packet forwarding hops by analyzing the network topology with nodes closer to the user for caching. Nodes at higher levels cache less popular content so that the content of most interest to users is cached for hits at the edge routers and the rest is stored at the upstream routers, improving the overall hit rate and cache space usage efficiency and reducing the number of hops to forward packets of interest for popular content.

3.1. Content Popularity Rating Model

3.1. Content Popularity Rating Model. In the content popularity rating model of this paper, the data content in the ICN is graded according to the popularity of the content, and the level indicates the importance of whether the node is cached, represented by the following: 0 to 3.0 means unnecessary caching, 1 means unknown level, 2 means cacheable, and 3 means high cache value. The content should be graded differently for different node popularity, because the scheme takes into account both the location of the caching node in the network topology. That is, the behavioural performance of the content on a communication path is depicted as shown in Figure 1.

Figure 1 shows part of the communication path in the network. Consumers 1 and 4 both request C_n and Consumer 1 also requests C_q. So, C_n is the more popular content and is cached by R_1. However, in R_3, the cache level of C_n drops to 1. Similarly, for R_2 and R_4, the highest levels are C_m and C_q requested by Consumers 2 and 3. In particular, Consumer 2 also requests C_p at the same time at a lower frequency, so the level of C_p is somewhat lower than C_m. $C_o$is stored at the R_5 node close to the producer, and since it is not stored at any of the edge nodes, the packet of interest will be forwarded to R_5.

3.1.1. Request Frequency Collector. Massive content access is the most important feature of ICNs. With the emergence of more and more large content providers and user-produced content applications, there is a trend towards diversification of content and applications on the network. The transparency of in-network caching in ICNs separates applications from caches, allowing different types of content from different content servers to be stored in the cache space of the same node. This makes the analysis of network caching systems very difficult. The idea of popularity hierarchy was explained in the previous section in conjunction with a diagram. Since the popularity of content varies over time, in order to achieve a classification for the popularity level, it should be dynamically adjusted according to its request frequency. Specifically, this section implements a packet-of-interest request frequency collector, which is used as a basis for differentiating the popularity of content entries stored by router nodes into different levels.

The popularity of the content reflects the user's interest in the content at the local time for a period of time. In order to ensure that the content popularity can truly

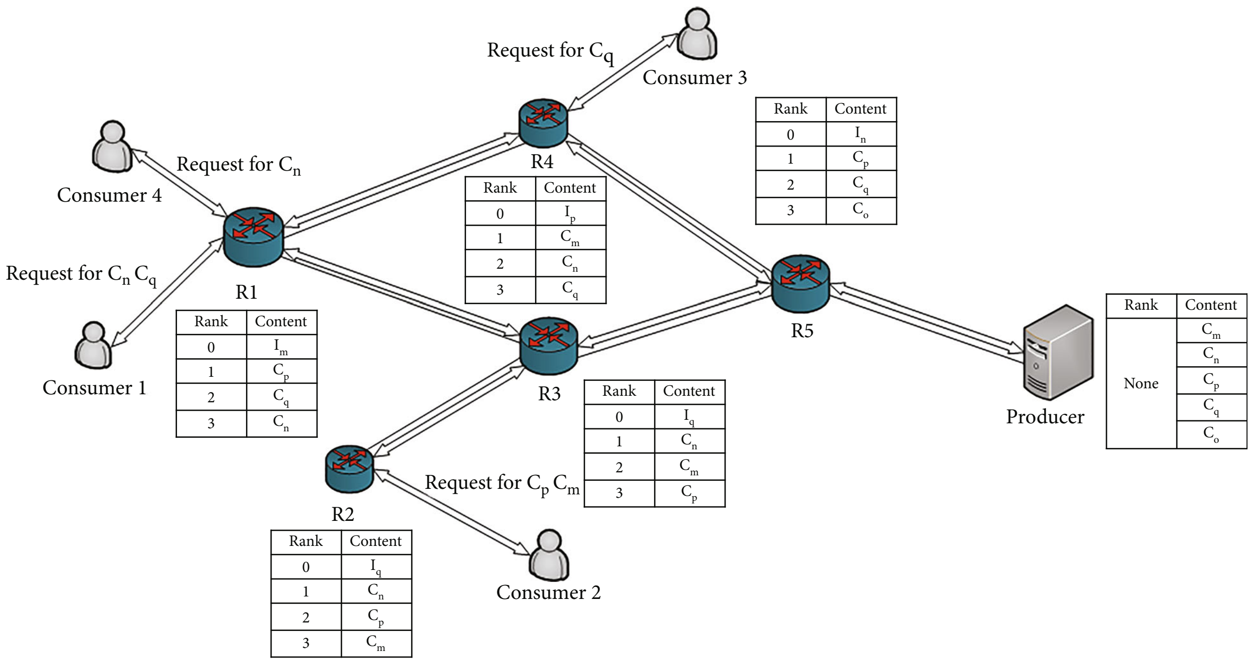

FIGURE 1: Interest packet forwarding process.

reflect the content request situation of the current network, each router node will regularly update the access frequency of the interest packet. Specifically, this scheme first defines a time period T. Then, each router node updates the request frequency of interest packet I_i corresponding to the requested content C_i at the end of each period, denoted by rF_i^j, as in

$$rF_{C_i}^{j} = \beta * MF_{C_i}^{T_j} + (1-\beta) * rF_{C_i}^{j-1}, \tag{1}$$

where j is the count index of a certain time period, indicating the jth time period. $MF_{C_i}^{T_j}$ is recorded to indicate the number of requests for packet of interest I_i collected by the router node in the jth period. $rF_{C_i}^{j-1}$ is the frequency of interest packet requests recorded in the previous time cycle, i.e., the $j-1$th period. And $\beta \in (0, 1]$ is a weighting factor. Obviously, β reflects the trend of tracking the change in request frequency. The larger the β, the faster the response to changes in the frequency of interest packet requests. However, using a large β value will also cause the observed request frequency to fluctuate more frequently as well. For this reason, it is set by default to 0.6 in this chapter, in the hope that it will reflect the trend in user preferences and thus consider future popularity.

3.1.2. Popularity-Based Cache Hierarchy. The set of stored content items is divided into 4 levels of different popularity. Consider a collection of content items M of size $|m|$ whose content popularity follows a Zip-f distribution. Thus, let rank be a random variable indicating the popularity level of the requested object, then rank (C_i) is the popularity level of content object C_i, which when it takes the values 0, 1, 2, or 3, respectively, denotes the meaning as shown in Table 1.

TABLE 1: Cached content collection popularity ranking.

Popularity rank	Implication
0	No cache necessity
1	Unknown prevalence
2	Consider to cache
3	High cache value available

Thus, rank $(C_i) = 3$ is the set of most popular content objects, i.e., they account for 40% of the total number of requests. Similarly, rank $(C_i) = 2$ is the set of objects that receive the next 30% of requests; while rank $(C_i) = 1$ is unknown for its popularity rating, they account for 20% and have a tendency to shift to a potentially higher popularity rating, as well as the possibility of swapping out content entries that belong to a rating of 2 or 3. Finally, the set of content entries with rank $(C_i) = 0$ accounts for 10%. It is worth noting that their state in the cache changes most erratically. Because if new content wins the storage competition when the cache fills up, they are the set that is to be replaced out first, but this does not indicate that they will never be cached by a node.

The structure of the hierarchy in the node cache is shown in Figure 2. For a set of content collections classified to the same popularity level, they will be sorted according to the observed frequency of interest requests. That is, there are actually two combinations of sorting here, the first by popularity level and the second by request frequency. Content collections that are downgraded from a higher level of popularity are stored in the next level of collections first comparing their popularity with the top entries in that level and then finding the right place to insert them, thus

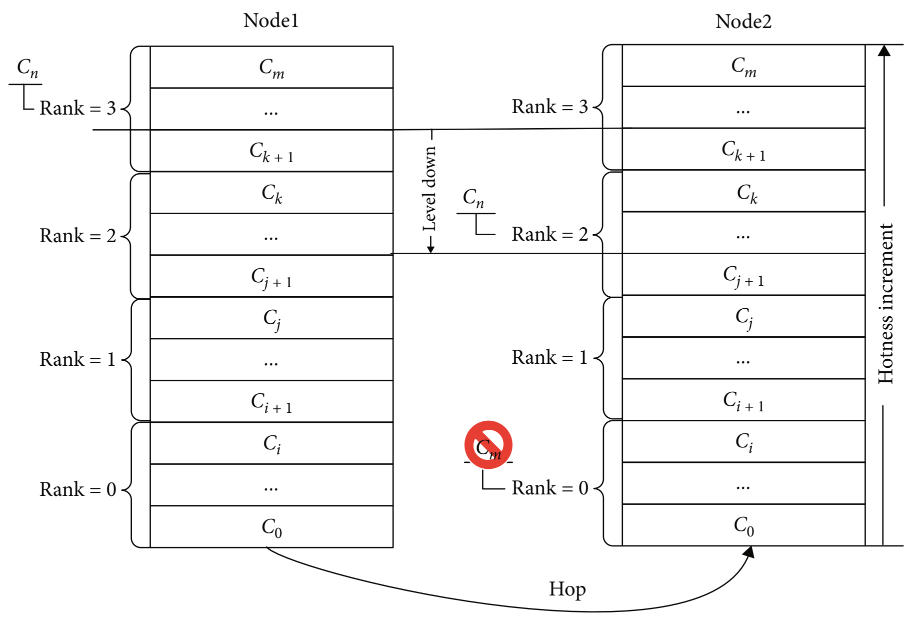

FIGURE 2: Distribution of popularity content segmentation.

distinguishing between frequently hit and frequently replaced content collections.

In Figure 2, the content popularity is increasing in rank. C_n being cached in the CS of node 1 has a rank value of 3, which is determined by the popularity of the content, as consumers connected to node 1 will have a higher frequency of requests for C_n, which will then have a higher popularity level according to the setting in the scenario. Obviously, the impact of this scenario for node 2 is that node 1 will not send interest packets for C_n from node 1 to node 2 when it satisfies the interest packet request for C_n, so the rank value of C_n becomes smaller in node 2. The set of contents stored in each node along the entire communication path is not identical; they are intersecting sets. So, after the CS in node 2 stores the new content, it expels C_m with a rank value of 0 from the storage space.

When a router node receives a request for content belonging to rank $(C_i) = 0$ and records its interest packet request frequency, if the cache is full, it will mark the interest packet with the topology node weight BeCentry(n) value and then forward it to the next-hop router node; the next-hop node compares this value with its own BeCentry(n) value based on this value, and if the next-hop BeCentry(n) value is higher than the source router's BeCentry(n) value, then the popularity level is raised at this next-hop node, and then, the decision to store the packet in the cache is based on the correlation analysis of its popularity level with the topology node weight.

The popularity ranking is a mapping of the consumer's interest in the content to the rank, and the individual content entries are ranked with the calculated interest packet request frequencies between them to facilitate the comparison of their popularity levels with the request frequencies. Once the interest packet request frequency is collected, if the interest packet cannot be responded to locally, the interest packet structure is reconstructed to include the request

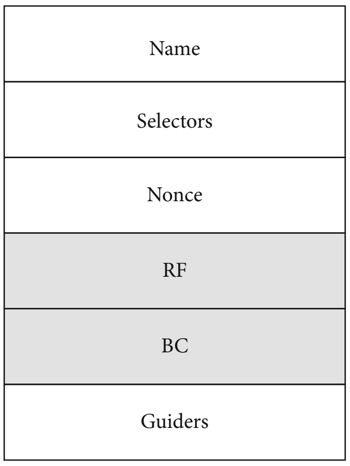

FIGURE 3: Modified interest package structure.

frequency rF_i^j collected at that node in the header information of the interest packet. Therefore, the interest packet needs to add the marker field RF indicating the request frequency, which is extracted for processing when the interest packet is received and added for updating when it is forwarded.

3.2. Topological Node Weights

3.2.1. Topological Node Importance Evaluation Method. The homogeneous caching scheme ignores the differences in the importance of nodes in the network topology and content distribution. Although this default scheme is simple and easy to implement, it does not make full use of the caching capacity of upstream nodes and does not distinguish the difference in caching capacity between core and edge nodes, which seriously affects the performance of the caching system.

```
Input: Node Received Interest packet Interest($C_i$)
Output: The ranking state and processing status
1:  if $C_i$ is in $CS$ then
2:  Parse header information from interest packages for BC, RF
3:     Calculate $\Delta BC$ according to (4-3)
4:  if $\Delta BC \geq 0$ then
5:  $MF_{C_i} \longleftarrow MF_{C_i} + RF$
6:  else
7:  $MF_{C_i} \leftarrow MF_{C_i} - 1$
8:     Forward $Data(C_i)$ in the reverse path, according to the pit interface
9:  return SUCCESS
10:     else
11:     rank ($C_i$) initialized as 1
12:     Parse header information from interest packages for RF
13:     if $Interest(C_i)$ is in $PIT$ then
14:     $RF \longleftarrow new\,RF + RF$
15:     Add RF to interest package, update $Interest(C_i)$ item in $PIT$
16:     Discard $Interest(C_i)$
17:     return SUCCESS
18:     end if
19:     end if
20:     if $Interest(C_i)$ in FIB then
21:     Add RF to interest package, update $Interest(C_i)$ item in $FIB$
22:     Forward $Interest(C_i)$ to the next hop
23:     return SUCCESS
24:     else
25:        Discard $Interest(C_i)$
26:     return FAIL
27:     end if
28:     Update the rank layout of the content according to equation (1)
```

Algorithm 1: The packet of interest processing for this strategy.

Therefore, to improve the performance of the ICN caching system, a heterogeneous allocation approach is needed, where nodes that play an important role in content distribution should be allocated a larger cache size. Therefore, network topology will necessarily have a significant impact on content distribution, but in many scenarios, the importance of nodes in the topology is not exactly the same as the importance of nodes in content distribution. Data paths on router nodes connecting multiple users contain a large number of different requests, and such nodes should be allocated more cache size so that they are weighted higher than paths connecting fewer users.

As will be explained next, the centrality of a node will be a measure of how widespread cached content is when served on that node. Considering network topology information to define metric node weights, it provides support for designing hierarchical caching strategies based on topology weights and expected popularity.

Node centrality is used to measure the number of times a node appears in the content delivery path [15]. Nodes with higher centrality values can access more content on the network. Limited by the size of the cache, the priority of cache node selection is proportional to the centrality; that is, the higher the centrality, the higher the score, and the more content the node should cache. Using this graph to represent a router network, the centrality of router node n is expressed as the following:

$$\text{BeCentry}(n) = \sum_{\forall s,t \in V \backslash n} \frac{\sigma_{st}(n)}{\sigma_{st}}, \tag{2}$$

where σ_{st} is the number of shortest path entries between s and t and $\sigma_{st}(n)$ is the number of shortest path entries between s and t through n.

The assumption made for the topology node weight-based part is that BeCentry(n) is calculated offline in advance for all routers, so that each router node knows its own BeCentry(n) value and marks this value when forwarding a packet of interest, i.e., by adding a BC field to the packet of interest. In the caching policy, router nodes receive requests and not only extract the RF field from the packet of interest to aggregate the content request frequency but also determine the popularity level of the requested content object. The BeCentry(n) value of the router node that originated the packet of interest request is also extracted from the packet of interest for comparison with the BeCentry(n) value of the current node, as shown in

$$\Delta\text{BC}(C_i) = \text{BeCentry}(n) - \text{BeCentry}\left(C_i^{r1}\right), \tag{3}$$

```
Input:Node Received Data packet $Data(C_i)$
Output: Processing status
1: if$C_i$ not in PIT then
2: Discard the $Data(C_i)$
3: return FAIL
4:       else
5: ifrank $(C_i) = 3$then
6:            if$CS$is Filled then
7:       Execute LFU policy within current rank,delete content in rank = 1
8: Insert $C_i$ into $CS$
9: end if
10:          else ifrank $(C_i) = 2$then
11:          Execute LFU policy within current rank,delete content in rank = 2
12:          else ifrank $(C_i) = 1$then
13:                Insert $C_i$ into $CS$in probability according to equation (4)
14:          elseifrank $(C_i) = 0$then
15:                continue
16:          Forward $Data(C_i)$ according to FIB
17:             return SUCCESS
18: end if
19:          end if
```

Algorithm 2: The packet of data processing for this strategy.

where $\Delta\text{BC}(C_i)$ represents the difference between the mesoscopic centrality of two router nodes, $\Delta\text{BC}(C_i)$ is the mesoscopic centrality of the current node n, and $\Delta\text{BC}(C_i)$ is the currently received mesoscopic centrality for content C_i from node $r1$. They may take on positive, negative, or zero values; if positive, it indicates that the current node is more important for content C_i; if negative, it indicates that node $r1$ has a stronger role in the topology; if zero, it indicates that both are of equal importance.

3.2.2. Correlation Analysis of Popularity and Topological Weights. The goal of the caching policy is to select a subset of router nodes in the transport path for specific content caching. The algorithm is based on the correlation of content popularity and topology importance. The two correlations show how well the two match from a content perspective and a topology perspective. The hierarchy of nodes for cached content should adapt to the different distributions of nodes in the network topology and be able to make dynamic adjustments on its own.

The correlation function used in the calculation of the correlation between popularity and topological weights calculates the absolute difference between the two variables. As expressed in

$$\text{COV}(\text{RF}, \text{BC}) = \text{Correlation}(\text{RF}, \text{BC}). \tag{4}$$

In this scenario, a popularity-based scheme is used to update the cache. As mentioned earlier, each packet carries a content popularity level rank on its transmission path. When a decision is made to add a new revenue content to a router whose cache is full, the router compares the frequency of requests for the new content with the rank of the content in the same popularity level in the cache. If the new content is requested more frequently, the content in the cache with a lower request frequency in the same popularity level will be ranked decentralized; this operation will only cause a redistribution of the content popularity level distribution in the content collection if the cache is not full; however, when the cache is full, a cache replacement operation will be triggered. Otherwise, the new content will be discarded.

Table 2: Basic experimental parameter settings.

Parameter name	Parameter values
Topology	GARR, GEANT, TISCALI, WIDE
Request rate	30 s^{-1}
Number of contents	100000
In-network cache size	[0.03,0.05,0.08,0.1]
Values of α	[0.8,1.2,1.6,2.0]
Int-domain time delay	2 ms
Out-domain time delay	30 ms
Number of prefilled caches	3000
Actual number of caches measured	6000
Branching factor	8

3.3. PT-Cache Specific Solution Design. To begin with, if a router node fails to respond to a particular packet of interest, the frequency of interest packet access to that node will be aggregated to the next-hop router node. In practice, two phenomena exist in ICN networks.

(1) The closer a router node is to the source server of a packet of interest, the better the chance that the

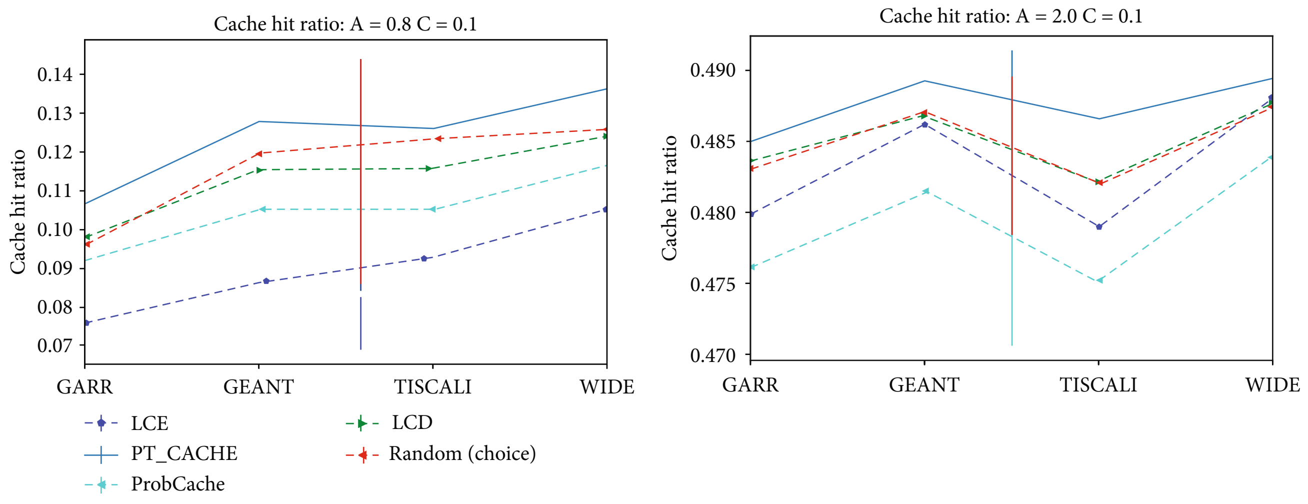

Figure 4: Cache hit rate for different α values in different topologies.

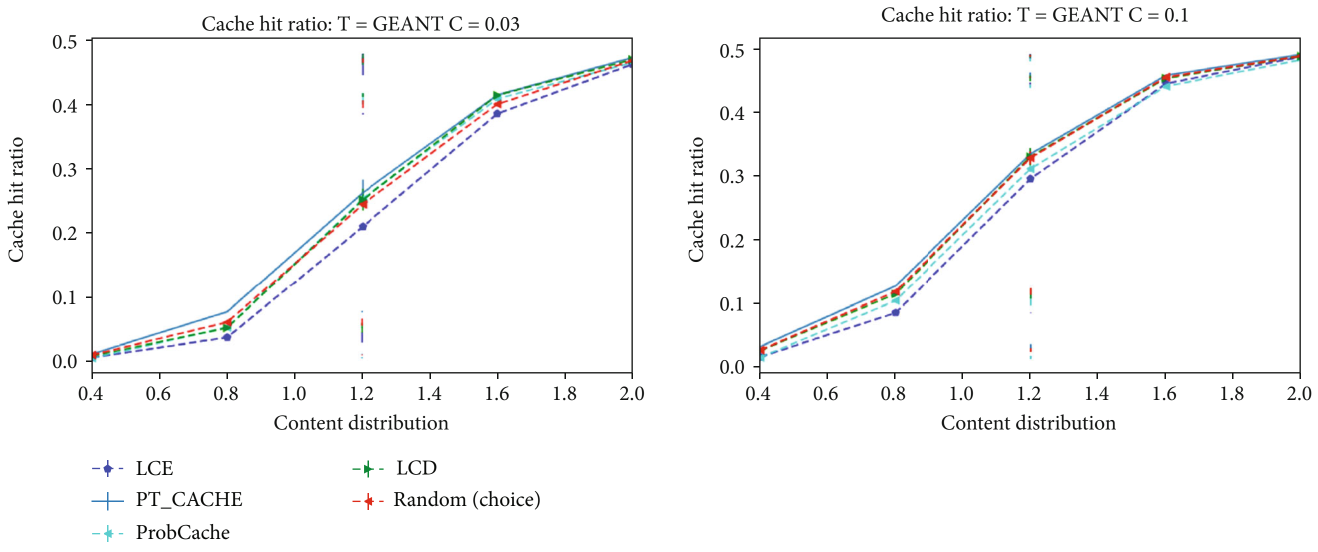

Figure 5: Cache hit rate for different in-network cache sizes under GEANT.

router node will aggregate the frequency of access from the packet of interest. In other words, the closer a router node is to the producer, the more likely it is to aggregate packets of interest sent from downstream

(2) The closer the packet is stored to the node requesting it, the fewer hops it can reduce the number of forwarding hops for the same request frequency. This is considered for the case where the packet of interest needs to be forwarded

By combining the two scenarios above, it can be seen to store popular content packets close to the consumer and less popular packets away from the consumer. In this way, packets with high popularity levels are maximised to preserve the number of forwarding hops, while more frequent requests are aggregated for content with average popularity levels. To this end, the following strategy is proposed.

(1) All packets arriving at the router compete for cache space based on popularity level and topology node weights

(2) The expected caching or noncaching of packets is calculated based on the correlation between the frequency of packet of interest requests measured at the router node and the topology node weights

In this process, packets with high popularity levels are considered first in the competition of nodes to be cached due to their higher request frequency. However, their caches at or near the edge router will directly satisfy the request. Packets of interest will be "blocked" at this node, thus

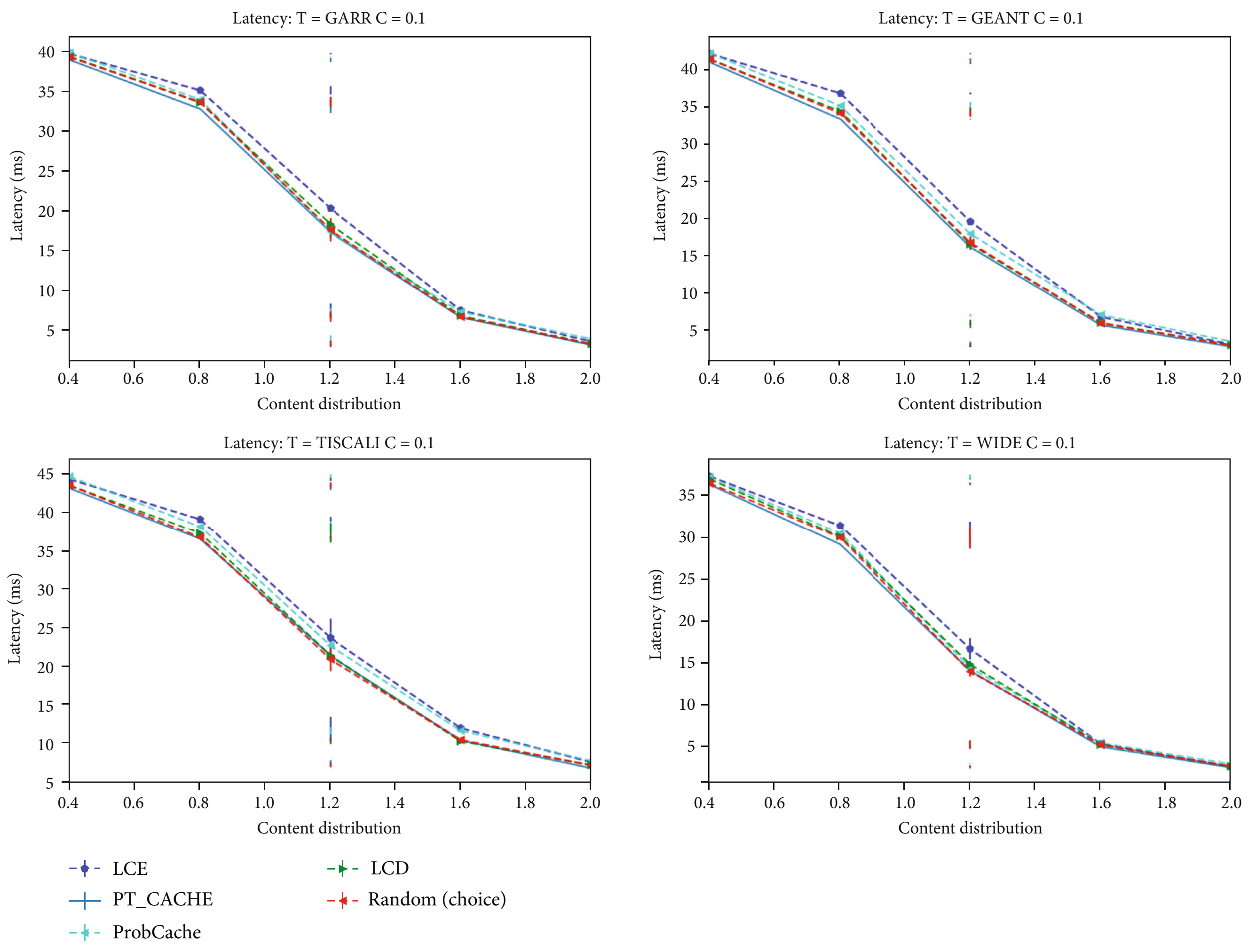

FIGURE 6: Average response delay for different topologies.

reducing their frequency of requests at routers close to the consumer. This allows the popularity level to change from router node to router node, so that content storage nodes are reasonably spread out in the network, rather than frequently replaced and rewritten at the same node. Packets with lower prevalence levels at edge nodes are able to boost the prevalence level height and thus gain caching opportunities when they are forwarded to upstream nodes by aggregating the request frequency. As a result, popular packets sink towards the consumer side and unpopular packets move closer to the producer side.

In order to implement the collection of interest packet request frequencies and the calculation of relative topological weights in the scheme, a modification to the interest packet structure is required, which modification is shown in Figure 3.

The RF and BC fields are added to interest packets; RF indicates the content popularity level of the interest packet on its source node and the frequency of interest packet requests, and BC is the mesocentricity of the node. The validity stems from the fact that as requested content is added to the cache, they are ranked up and down in order of content popularity. By adopting this strategy, diversity in the cache repository can be achieved, as less popular content can be cached on more distant nodes. Algorithm 1 shows the packet of interest processing for this strategy.

The solution does not change the overall structure of the packet, but does so because in an information-centric network, where content is the object of distribution, all copies of the same content in the network should be identical. Caching in different nodes is simply a copy of the content, and modifying the packet not only defeats the purpose of the clearinghouse network design but also bloats the packet and takes up valuable network resources during communication. Algorithm 2 shows the packet of data processing for this strategy.

4. Experimental Results and Analysis

4.1. Experimental Program. This paper implements the proposed PT-Caching scheme on Icarus [16]. The hardware environment on which the platform runs is the same as in Section 3 and will not be elaborated on. Icarus is a python-based caching simulator for ICN-based architectures (focusing on CCNX and NDN), using named content, a request-response model (e.g., interest and content

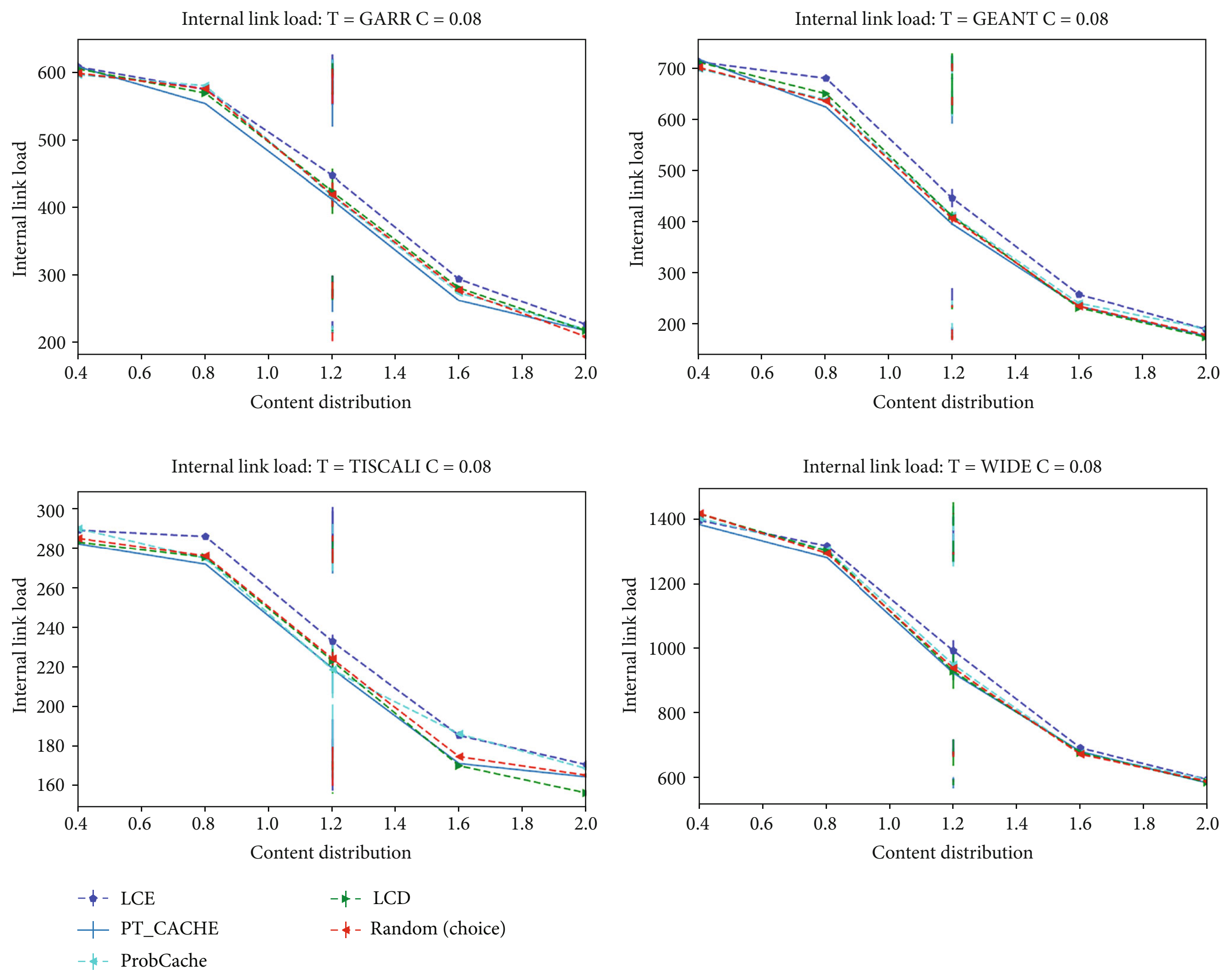

Figure 7: Link load for different topologies.

request), and supporting various evaluation protocols. The proposed solution is compared in performance with the cache placement policy (location of placed content copies) supported by Icarus.

Four real network topologies were used in the experiments, namely, GARR (Italian national university and research computer network), GEANT (European data network for the research and education community), TISCALI (Italian telecommunication network), and WIDE (from Japan). The performance of the cache hit rate, average latency, and link traffic for each scenario was evaluated for different content popularity distribution parameters α and total network cache capacity, and the basic parameters of the simulation were set as shown in Table 2.

In Table 2, 30 s^{-1} is the number of requests per second (over the whole network) that belongs to the experiment parameters. That is to say, 30 s^{-1} is set to thirty requests per second.

This section compares the proposed scheme with the LCE, LCD, ProbCache, and Random schemes setting different network topologies, α values to compare cache hit rates, average response latency, and link traffic. The following are the basic ideas of the four schemes.

LCE: in this method, packets are cached at each node of the path as they are forwarded downstream. This means that the content is cached at each node along the path.

ProbCache: this policy reduces the redundancy of cached content by probabilistically caching content on the way.

Random: in this caching policy, content can be cached on any of the downstream nodes. The content is cached on a particular downstream node which is chosen at random.

4.2. Analysis of Results

4.2.1. Cache Hit Rate. Figure 4 shows the hit rates for the four network topologies for different α fetching values. α values are used to generate the Zip-f distribution of content requests and must be positive. The larger the value, the more skewed the distribution of content popularity. Therefore, as the α value increases, the hit rate of all scenarios improves. PT-Cache shows a high hit rate in all topologies, and the performance advantage is especially evident in the GEANT topology. This is since the GEANT topology has a more pronounced hierarchical feature compared to other topologies, which is suitable for the content popularity grading and extraction of node topological weights in the proposed scheme. The hit rate performance of each scheme in the GEANT topology for different network cache sizes can be observed in Figure 5. As the content distribution popularity value increases, the hit rates of all schemes improve, with LCD and PT-Cache having the best hit rates and LCE being the worst, and this scheme will have a large amount of cache redundancy in the path.

4.2.2. Average Latency. Average latency quantifies the duration from requesting a file to delivering it and is an intuitive representation of the user's network experience. As the use of streaming applications grows, response time becomes increasingly important. The average hop count is also a factor in the average latency, so the average latency is used to measure whether the average hop count has decreased. Shown in Figure 6 is the average response latency for each topology at different α values. At α values less than 0.8, each strategy exhibits a high average delay with a small difference in values. However, when the value of α is larger, within the interval $[0.8, 1.6]$, the average latency of each strategy tends to decrease substantially with the concentration of content popularity. However, when the value of α exceeds 1.6, when the content popularity is very concentrated, the proposed scheme has no obvious advantage in this case, the average latency of all the schemes decreases at a slower rate, and the performance of each strategy is relatively similar.

4.2.3. Link Load. The amount of data passing through the transmission path at each time is defined as link traffic. In an ICN, this metric is positively correlated with redundant traffic. Analyzing link occupancy helps to ensure the quality of service in the network. Figure 7 shows that the trend of link traffic under the influence of the α value is very similar to the first two metrics. On the one hand, with $\alpha = 1.6$ as a threshold, the performance advantage of the PT-Cache becomes insignificant. The PT-Cache policy has the lowest load on the link traffic in the range $[0.4, 1.6]$, due to its hierarchical treatment of the cached content in the nodes on the path, which better eliminates redundancy. On the other hand, the higher prevalence of content that can be cached to the nodes brings a high hit rate, which reduces request retransmissions.

The results of these experiments show that the PT-Cache method performs better under the three metrics of cache hit rate, average latency, and link traffic, even under different topologies, with the best performance under α values of $[0.8, 1.6]$, and its advantage is no longer obvious beyond 1.6. This is because it is difficult to distinguish clearly between the levels of content distribution, which is like that of a dedicated server, and the other solutions also show a significant performance improvement under such conditions. Therefore, the solution is better suited to be deployed in networks with a high diversity of content and a high degree of dynamism to take advantage of it.

5. Conclusions

High cache hit ratio, low data retrieval cost, and low user latency are the goals pursued in designing cache mechanisms. This section designs a caching method PT-Cache based on popularity and topology weighting. For different nodes, the content they are interested in is different, so for different nodes, the content to be stored preferentially is also different. Therefore, content popularity and packet hop-saving capability are the basis for packets competing for caching opportunities. Adopting this method helps router nodes to distribute content more reasonably and reduce cache redundancy in the network. Finally, PT-Cache is compared with other caching schemes with different topologies through simulation, which shows that PT-Cache can achieve better performance under different topologies and parameters, and the scheme can be effectively applied to various social network type web apps.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant Nos. 61831007, 61971154, and U21B2019) and the Fundamental Research Funds for the Central Universities (Grant No. 3072022CF0601).

References

[1] M. Conti, A. Gangwal, M. Hassan, C. Lal, and E. Losiouk, "The road ahead for networking: a survey on icn-ip coexistence solutions," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2104–2129, 2020.

[2] Y. Miao, Y. Wu, and W. Wei, "Co-clustering of multi-entities sparse relational data in mi-croblogging," *Journal on Communications*, vol. 37, no. 1, pp. 151–159, 2016.

[3] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD interconnection of LRU caches and its analysis," *Performance Evaluation*, vol. 63, no. 7, pp. 609–634, 2006.

[4] L. Ramaswamy and L. Liu, "An expiration age-based document placement scheme for cooperative web caching," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 5, pp. 585–600, 2004.

[5] G. Zhang, B. Tang, X. Wang, and Y. Wu, "An optimal cache placement strategy based on content popularity in content centric network," *Journal of Information & Computational Science*, vol. 11, no. 8, pp. 2759–2769, 2014.

[6] M. Bilal and S.-G. Kang, "Time Aware Least Recent Used (TLRU) cache management policy in ICN," in *16th International Conference on Advanced Communication Technology*, pp. 528–532, Pyeongchang, Korea (South), 2014.

[7] M. D. Ong, M. Chen, T. Taleb, X. Wang, and V. C. M. Leung, "FGPC: fine-grained popularity-based caching design for content centric networking," in *Proceedings of the 17th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems - MSWiM '14*, pp. 295–302, Montreal, Canada, 2014.

[8] C. Pu, "Pro$^{\mathrm{NDN}}$: MCDM-based interest forwarding and cooperative data caching for named data networking," *Journal of Computer Networks and Communications*, vol. 2021, 2021.

[9] C. Zhang and H. Wang, "Cooperative caching method based on neighbor node content and space," *International Core Journal of Engineering*, vol. 7, no. 6, pp. 88–96, 2021.

[10] Q. Zheng, J. Zhang, R. Wu, H. He, X. Tan, and L. Yuan, "An ICN cache pricing mechanism based on non-cooperative game model of users and advertisers," in *2020 3rd International Conference on Hot Information-Centric Networking (HotICN)*, pp. 77–83, Hefei, China, 2020.

[11] H. Liu and R. Han, "A hierarchical cache size allocation scheme based on content dis-semination in information-centric networks," *Future Internet*, vol. 13, no. 5, p. 131, 2021.

[12] V. S. Shekhawat, A. Vineet, and A. Gautam, "Efficient content caching for named data network nodes," in *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pp. 11–19, Houston, United States, 2019.

[13] K. H. Chiu, J. M. Wang, A. M. Abdelmoniem, and B. Bensaou, "A two-tiered caching scheme for information-centric networks," in *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*, pp. 1–6, Paris, France, 2021.

[14] M. Alhowaidi, D. Nadig, B. Hu, B. Ramamurthy, and B. Bockelman, "Cache management for large data transfers and multipath forwarding strategies in named data networking," *Computer Networks*, vol. 199, article 108437, 2021.

[15] B. Nour, H. Khelifi, H. Moungla, R. Hussain, and N. Guizani, "A distributed cache placement scheme for large-scale information-centric networking," *IEEE Network*, vol. 34, no. 6, pp. 126–132, 2020.

[16] L. Saino, I. Psaras, and G. Pavlou, "Icarus: a caching simulator for information centric networking (icn)," in *Proceedings of the Seventh International Conference on Simulation Tools and Techniques*, pp. 66–75, Lisbon, Portugal, 2014.