IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 10, 2021, accepted March 27, 2021, date of publication April 6, 2021, date of current version April 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3071427

A Cache Placement Strategy Based on Entropy Weighting Method and TOPSIS in Named Data Networking

YIQI GUI AND YONGKANG CHEN
College of Information Engineering, Yangzhou University, Yangzhou 225009, China
Corresponding author: Yiqi Gui (gigi_13688@163.com)

This work was supported in part by the Natural Science Foundation of Jiangsu Province under Grant bk2015045, and in part by the Research Start-Up Fund of Yangzhou University under Grant 137010481.

ABSTRACT Named data networking (NDN) aims to change the traditional content delivery method and caching by router nodes caching and participating in forwarding. NDN-caching can reduce the expected flood of global data traffic by providing cache storage at intermediate nodes for transmitted content objects, making data broadcasting in an efficient way. In this paper, a novel caching strategy based on entropy weighting method and TOPSIS is proposed for efficient content dissemination to improve NDN's in-network caching performance. Firstly, the consumer's request process is modeled by the entropy weighting method and TOPSIS to obtain the best cache node for the cache object according to the real-time status of the node. Secondly, two cache replacement algorithms (composed of an active cache replacement algorithm and a passive cache replacement algorithm) are proposed to reduce the cache redundancy on the delivery path and improve the utilization of data packets in the nodes. Finally, an effective cache mechanism and data packet migration scheme are proposed to further improve the cache performance according to the different types of cache nodes. The performance evaluation shows that the proposed scheme performs better in terms of cache hit rate, latency, and link load compared with some existing strategies.

INDEX TERMS Information-centric networking (ICN), named data networking (NDN), TOPSIS, in-network caching, on-path caching.

I. INTRODUCTION

The main function of the Internet is to satisfy consumers' needs for connectivity and resource sharing. Content acquisition services represented by video distribution and file downloads have become mainstream application requirements on the Internet in the last decade. According to Cisco Annual Internet Report [1], the total number of Internet users will increase to 5.3 billion (66% of the global population) in 2023, which is higher than 3.9 billion (51% of the global population) in 2018. The growth of Internet users makes the total amount of Internet video services also increase. The wide applications of CDN (Content Distribution Network, CDN) [2] and P2P (Peer-to-Peer, P2P) [3] technologies based on TCP/IP traditional network architecture have alleviated the network problems caused by mass content distribution to a certain extent. However, so many disadvantages of traditional network architecture have not fundamentally solved in packet distribution, such as low security, insufficient flexibility, and poor scalability [4], which makes it difficult to satisfy new network requirements in the future [5]. To improve the quality of network services, researchers have proposed the Information-centric networking (ICN) [6] architecture, including Data-Oriented Network Architecture (DONA) [7], Network of Information (NetInf) [8], Publish-Subscribe Internet Routing Paradigm (PSIRP) [9], Named Data Networking (NDN) / Content-Centric Networking (CCN) [10] and so on. Although these ICN network architectures design are different, most of them aim to transform the existing content acquisition mode to solve the key issues of the TCP/IP network architecture.

Named Data Network (NDN) is one of the famous ICN-type network architectures, which achieves efficient content distribution by constructing a unified

The associate editor coordinating the review of this manuscript and approving it for publication was Guangjie Han.

network architecture. The efficient content distribution performance of NDN depends on the in-network caching function. In-network caching of NDN is provided to the network as an infrastructure service, which can effectively improve the quality of network services to maximize the huge benefits for consumers and Internet service providers [11]. Furthermore, in-network caching can be combined with Vehicular Ad Hoc Networks [12], software defined networking (SDN) [13], Fog Computing [14], blockchain [15], Internet of Things (IoT) [16], [17] and so on to implement flexible and efficient network services.

The caching strategies mainly include cache placement algorithm, cache replacement algorithm, and cache resource allocation algorithm in NDN. Cache placement algorithm is the key to improve the performance of caching strategy, so cache placement algorithm has been studied most at present [18]. To this end, we proposed A Cache Placement Strategy Based on Compound Popularity (abbreviated as CBCP) [19] by using global request information in our previous research. CBCP only includes a cache placement strategy, and also requires a small amount of communication overhead between nodes to obtain global data. To make more reasonable use of limited cache resources and avoid the communication overhead of obtaining global request information, we consider the content popularity within a single node and the changes in the state of the node to design a caching strategy instead of considering the global request information in this paper. The caching strategy includes cache placement strategy and cache replacement strategy. The main contributions of this paper are summarized as follows:

1) We analyzed the real-time status of nodes and the popularity of content within a single node in detail and used entropy weighting method and TOPSIS to model the process of consumers requesting content. The optimal cache node of the cache object can be reasonably calculated by this model.

2) We also propose a novel compound cache replacement strategy, which consists of active and passive cache replacement algorithms. The cache replacement algorithms can effectively improve the utilization of data packets and reduce the cache redundancy in the network. To further improve the reuse rate of data packets, we also propose an effective cache mechanism and packet migration scheme for different types of cache nodes.

3) By using Icarus simulator to conduct large-scale simulation experiments in the Tiscali-3257 (pan-European commercial ISP) topology, we evaluated and analyzed the reasons for the performance differences between the various caching strategies in detail. In the simulation experiment, the proposed scheme has achieved good performance in the three evaluation indexes of cache hit rate, latency, and link load.

The rest of the paper is organized as follows: Related work is discussed in Section II. The system model is described in Section III. Section IV describes the proposed scheme in detail, including the calculation of cache location and cache replacement strategy. The experimental results are presented in Section V. Section VI concludes this paper and provides an outlook for future work.

II. RELATED WORK

The inherent in-network caching function of NDN is a revolution in network architecture. It can overcome the issues arising in the current internet architecture. Moreover, it has the potential to significantly reduce the transmission delay and traffic load in the next generation network [20], [21]. However, it is difficult to decide which content needs to be cached at which location to produce effective and efficient results. Therefore, several cache management strategies [22]–[32] have been developed. It is not clear yet which caching mechanism is the most ideal for each situation. According to the following classification basis, the existing caching strategies of NDN can be divided into different categories.

A. CACHE PLACEMENT LOCATION

According to whether the cache location of cache objects is related to the delivery path of consumers, cache policies can be divided into on-path caching and off-path caching [33].

In on-path caching, researchers aim to find suitable cache locations on the delivery path for cache objects. LCE (Leave Copy Everywhere, LCE) [22] is the default caching strategy in NDN, which caches content objects at all on-path router nodes. LCE reduces the diversity of cached content, resulting in high cache redundancy in the network. Laoutaris *et al.* proposed LCD (Leave Copy Down, LCD) [23], which caches the content objects at the downstream neighbor node of the hit node along the delivery path. It takes a long time for LCD to gradually cache popular content to routing nodes close to consumers, but the cache redundancy in the network is not reduced. Chai *et al.* [24] proposed CL4M (Cache Less for More, CL4M), which caches content objects according to the betweenness centrality of router nodes on the delivery path. CL4M can improve the cache utilization of nodes with high betweenness centrality, but frequent cache replacement operations cause greater pressure on these nodes. Psaras *et al.* proposed ProbCache (Probabilistic caching, ProbCache) [25] which caches content at on-path nodes with a probability. ProbCache only considers the location and capacity attributes of cache nodes when caching content objects, but does not consider the content popularity, which leads to the spatial distribution of popular content not becoming uniform and reasonable. Nguyen *et al.* [26] designed a progressive popularity-aware caching scheme (PPCS), which designed a cache placement strategy and an autonomous replacement strategy to replace LRU according to the sequential playing rule of video blocks to improve the VOD performance in a tree topology. Wu *et al.* [27] proposed a heuristic probabilistic caching method, which calculated the cached probability of content based on factors such as cache revenue and content heat. Zhang *et al.* [28] proposed a collaborative caching scheme based on popular content awareness and tracking, which can cache content objects at specified positions on the

delivery path according to content popularity and the caching ability of nodes. The characteristic of off-path caching is that the cache location of the cache object is not limited to the delivery path, and the cache object can be cached anywhere in the network. Saino *et al.* [29] designed five different hash-routing schemes using hash routing as the basis of cache strategy, which can effectively utilize the cache in the network without the router node maintaining the content information. Yang *et al.* [30] proposed a lightweight regional cache collaboration method, which can share popular cache data between small-scale caches with the least exchange of cache information, especially for hierarchical named ICN. Zhang *et al.* [31] designed a novel hierarchical active caching method by using non-negative matrix factorization technology to predict consumers' preferences. Moreover, this caching strategy considers the mobility of vehicle consumers. Hua *et al.* [32] combined the Internet of Things and ICN to propose a scheme based on fog cluster, which uses router nodes and consumer devices to cache content closer to the edge network. In addition, the scheme also uses nodes near the content delivery path to cache content objects.

B. CACHE COOPERATION

According to the degree of cache cooperation, existing NDN caching strategies can also be divided into non-cache cooperation, explicit cache cooperation, and implicit cache cooperation. In the non-cache cooperation scheme, router nodes make caching decisions independently according to local caching policies without any mutual cooperation. LCE is a typical non-cache cooperative caching strategy.

Explicit cache cooperation makes caching decisions based on global information such as network topology and global content popularity to reduce content placement redundancy and make good use of network caching resources. Since most off-path caching strategies require different levels of communication overhead, most of them also belong to explicit cache cooperation, such as the above-mentioned literature [29]–[32]. It is worth noting that literature [29] needs less information interaction, which is a special case among many explicit cache collaborations.

Implicit cache cooperation relies on some additional information (such as content popularity, local routing node information, probability, etc.) to make caching decisions, which is neither as easy to cause a large amount of cache redundancy as non-cache cooperation nor as explicit cache cooperation requires a lot of global information to make caching decisions. Compared with explicit cache cooperation, implicit cache cooperation can also reduce the overhead of inter-node and global control, but it may reduce the efficiency of data usage and cause a certain drop in cache performance. Since the caching strategy on the path requires only a small amount of communication overhead, they basically belong to Implicit cache cooperation. Thus, LCD, CL4M, ProbCache, etc. in the above literature belong to implicit cache cooperation.

In summary, in terms of sorting by cache placement location, off-path caching can improve the utilization of content information, but it often involves a large amount of information interaction between nodes, and collecting and disseminating such information leads to huge communication overhead, thereby reducing cache performance. On-path caching follows the original design principle of ICN, so most caching strategies adopt this mechanism. In addition, on-path caching can also improve the stability of SDN with only a single controller [34]. In terms of sorting by cache cooperation, Non-cache cooperation is likely to cause cache redundancy. Explicit cache cooperation usually requires the exchange of a large amount of control information, and implicit cache cooperation requires a little control information to make caching decisions and can reduce content redundancy at the same time. In this paper, we only consider the content popularity within a single node and the real-time status changes of the nodes to design an on-path caching strategy and adopt implicit cache cooperation to minimize the communication overhead between nodes. Besides, we also designed a novel compound cache replacement strategy to improve the utilization of data packets and reduce the cache redundancy in the network.

III. SYSTEM MODEL

The system model is shown in Fig. 1. The set of all router nodes denote as $V = \{v_1, v_2, \cdots, v_i\}$ and each node $v_i \in V$ has a unique id i. $E = \{E_1, E_2, \cdots, E_j\}$ is denoted as the set of all edge nodes and $E \in V$. Unlike the traditional NDN model, each node (like v_1 in Fig. 1) is added with a Cache Information Table (CIT) and a Node Information Table (NIT). CIT is used to record the cache hit count from the contents stored in CS. NIT is used to record the node's betweenness centrality (BC), the number of cache replacements (CR), and the available cache capacity (ACC). In particular, the BC of each node can be obtained in advance and recorded in the node, which is a fixed value. The data structure of NIT is shown in Fig. 1.

In our scheme, the data structure of the original interest packet and data packet are modified. The introduction of the modified interest packet and data packet will be given in Section IV in detail. The content popularity is modeled by the Zipf function [35]. All router nodes can cache the delivered content and have the same cache capacity. Content block C is a basic cache unit. All content objects have the same size and are evenly distributed in all providers. Consumers' request paths are the shortest paths which are calculated by the Dijkstra algorithm to minimize the request-response time. The important abbreviations and symbols used in this paper are listed in Table 1.

IV. CACHE PLACEMENT STRATEGY BASED ON ENTROPY WEIGHING METHOD AND TOPSIS

The main objectives of our scheme are: (*i*) obtain cache location for popular content by using entropy weighting method and TOPSIS; and (*ii*) two novel cache replacement algorithms

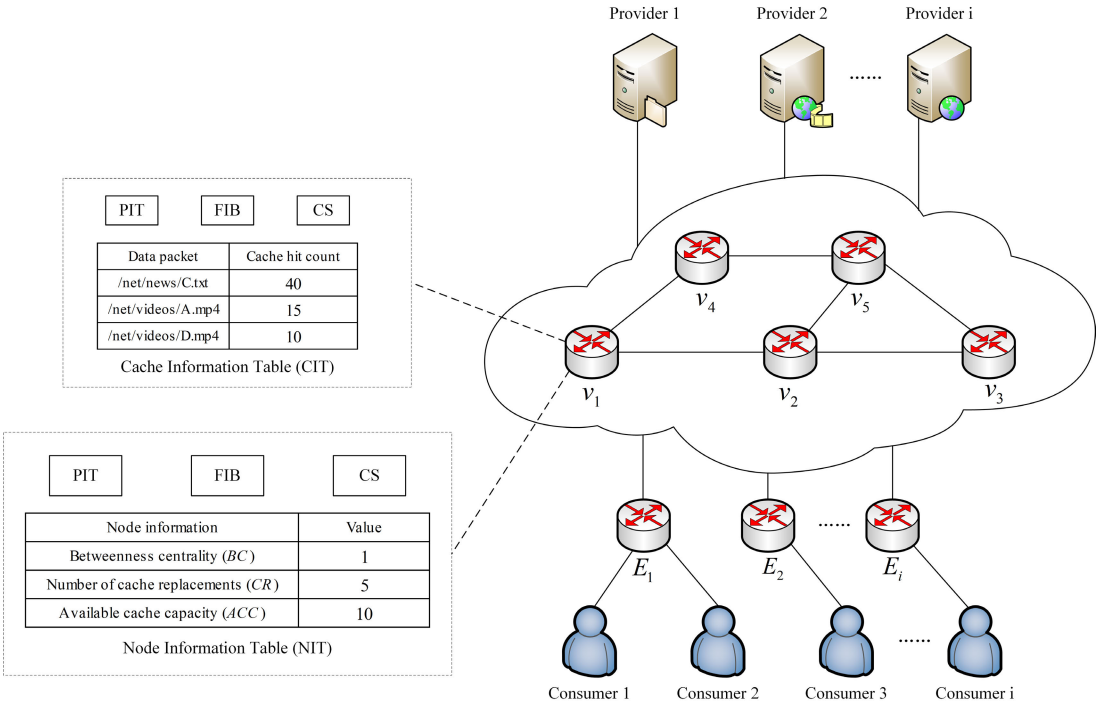


FIGURE 1. System model.

TABLE 1. Symbols used to describe the model.

Symbols	Description
V	The set of all router nodes
E	The set of edge nodes
v_{hit}	Cache hit node
v_c	Cache node
N_{E_j}	Number of received requests for edge node E_j
B	Number of cached contents of a single node (i.e. cache capacity of a node)
D_{v_i}	Degree of node v_i
$N_{v_c}^C$	Cache hit count of content C at node v_c
$N_{v_c}^{median}$	The median of the cache hit count of all content objects at node v_c
L_{v_c}	The content with the lowest cache hit count at node v_c
v_c^u	An upstream node one hop away from node v_c along the delivery path
v_c^d	A downstream node one hop away from node v_c along the delivery path
$r_{v_i}^k$	The ranking range of cache hit count of content k at node v_i

are used to effectively improve data packet utilization and reduce cache redundancy.

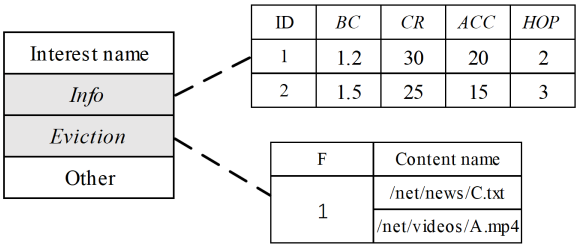
A. ENTROPY WEIGHTING METHOD AND TOPSIS

In NDN, the purpose of the on-path caching strategy is to select an appropriate caching node for the data packet from a limited number of nodes with different states on the delivery path. The caching decision problem in NDN can be modeled as a selection problem of a limited number of evaluation objects. To solve this problem, we use the entropy weighting method and TOPSIS to model cache decision to select the best cache node for cache object, based on four indexes of each node: *BC*, *CR*, *ACC*, and the number of hops from the current node to the consumer (*HOP*). The entropy weight method is an objective method to give weight, which gives weight to each index through the information uncertainty of each index. TOPSIS is a commonly used comprehensive evaluation method, which can avoid the subjectivity of data, does not require an objective function, and can describe the comprehensive influence of multiple indexes. Furthermore, TOPSIS has no strict restrictions on data distribution, sample size, and indexes. It is not only suitable for small sample data but also suitable for large systems with multiple evaluation units and multiple indexes, which is more convenient and flexible. Therefore, the entropy method and TOPSIS can effectively use the node's cache resources according to the node's state information and select the appropriate cache node for the cache object.

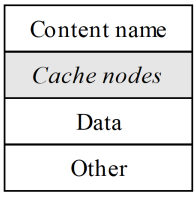
In our proposed scheme, the selected four indexes can describe node status information intuitively. Specifically, the *BC* is a measure of the importance of nodes in the network topology. If the *BC* of a node is relatively large, more paths will pass through the node, which makes the node face greater pressure. The *CR* denotes the number of cache replacements for a node in a period of time. Nodes with a large *CR* also face the problem of excessive node pressure, which reduces

node performance. Balancing the pressure between nodes can effectively improve the performance of the whole network. The *ACC* denotes the cache utilization efficiency of a node, and the smaller the *ACC* of a node, the higher the cache utilization rate of a node. The *HOP* denotes the distance between the consumer and the cache node. If the content object is cached at a node close to the consumer, the latency of the consumer will be greatly reduced, thereby improving the consumer experience. Thus, to improve the cache utilization rate of nodes and reduce the latency and node pressure, we aim to find the node with low *BC*, low *CR*, low *HOP*, and large *ACC* for caching content objects. In the proposed scheme, the *BC* of each node is a fixed value calculated in advance, and the remaining three indexes are dynamic which can be obtained from interest packets and the NIT of router nodes.

In some previous studies, researchers have expanded the structure of interest packages and data packages to varying degrees [27], [28]. To make the scheme to be successfully implemented, we modify the data structure of the original interest packet and data packet. In Fig. 2 (a), two fields are added to the interest packet: *Information* and *Eviction*. The *Information* field is a table, which records the ID of each node on the request path and the corresponding four indexes information (i.e. *BC*, *CR*, *ACC*, and *HOP*) of each node. The *Eviction* field is a list, which is used for the cache replacement strategy of the proposed scheme. The introduction of the *Eviction* field will be given in Section III in detail. In Fig. 2 (b), only the *Cache nodes* field used to record the ID of the cache node is added to the data packet.



(a) interest packet



(b) data packet

FIGURE 2. Extended structure of interest packet and data packet.

To reduce the computational overhead, entropy weight method and TOPSIS are used to calculate the cache location only for the popular content in router nodes. According to the principle of Zipf distribution (i.e. the top 20% of content requests account for 80% of the total number of requests), popular content in a single router node is defined as the top 20% of the content. The details of entropy weighting method and TOPSIS is listed as the following.

1) CONSTRUCT A STANDARDIZED MATRIX

The information used to construct the original decision matrix can be obtained from the *Information* field in interest packet. The original decision matrix is as follows:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix} \tag{1}$$

Matrix A indicates that the number of evaluation objects (i.e. evaluation objects are the router nodes between the consumer and cache hit node v_i) is n, and the number of each evaluation object's evaluation indexes is m. In this paper, evaluation indexes are *BC*, *CR*, *ACC*, and *HOP*. It is worth noting that not all of the indexes are positive indexes. Thus, all indexes need to be transformed into positive indexes. The calculation formula is as follows:

$$x_{nm} = \begin{cases} \max\{a_{1m}, a_{2m}, \cdots a_{nm}\} - a_{nm} \\ (m \text{ is a negative index}) \\ a_{nm} \\ (m \text{ is a positive index}) \end{cases} \tag{2}$$

The original matrix A is transformed into a positive matrix X by formula (2). Positive matrix X is as follows:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix} \tag{3}$$

Matrix X needs to be standardized due to the different dimensions of the four indexes. The formula used to eliminate the influence of different dimensions is as follows:

$$z_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{n} x_{ij}^2}} \\ s.t.\ \ i = 1, 2, \cdots, n; \quad j = 1, 2, \cdots, m \tag{4}$$

The matrix X is transformed into a standardized matrix Z by formula (4). Standardized matrix Z is as follows:

$$Z = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1j} \\ z_{21} & z_{22} & \cdots & z_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ z_{i1} & z_{i2} & \cdots & z_{ij} \end{bmatrix} \tag{5}$$

2) CALCULATE THE WEIGHTS OF DIFFERENT INDEXES

According to the definition of information entropy, the formula for calculating the information entropy e_j of the jth index is as follows:

$$e_j = -\ln \frac{1}{n} \sum_{i=1}^{n} p_{ij} \ln(p_{ij}) \tag{6}$$

$$s.t.\ p_{ij} = \frac{z_{ij}}{\sum_{i=1}^{n} z_{ij}} \tag{7}$$

If the value of p_{ij} is 0, then the value of e_j is also 0, which means that the information entropy of index j is 0. After obtaining the information entropy of each index, we can assign weights to each index according to the information entropy. The weight calculation formula of each index is as follows:

$$w_j = \frac{1 - e_j}{m - \sum_{j=1}^{m} e_j} \tag{8}$$

3) CALCULATE THE EVALUATION SCORE OF THE NODE

After obtaining the normalization matrix Z and the weight w_j of each index, a weighted normalization matrix Y for evaluating the scores of each node can be constructed. Weighted normalization matrix Y is as follows:

$$Y = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1j} \\ y_{21} & y_{22} & \cdots & y_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ y_{i1} & y_{i2} & \cdots & y_{ij} \end{bmatrix}$$
$$s.t.\ y_{ij} = w_j z_{ij} \tag{9}$$

According to the weighted normalization matrix Y, the positive ideal solution Y^+ and the negative ideal solution Y^- can be obtained, and the calculation formula is as follows:

$$Y^+ = (Y_1^+, Y_2^+ \cdots, Y_j^+)$$
$$s.t.\ Y_j^+ = \max(y_{ij}) \tag{10}$$

$$Y^- = (Y_1^-, Y_2^- \cdots, Y_j^-)$$
$$s.t.\ Y_j^- = \min(y_{ij}) \tag{11}$$

Finally, we can calculate the evaluation scores S_i of each node, and the node with the highest evaluation score will be considered as the cache node v_c. The calculation formula of S_i is as follows:

$$S_i = \frac{d_i^-}{d_i^- + d_i^+} \quad (i = 1,\ 2, \cdots, \mathrm{n}) \tag{12}$$

$$d_i^+ = \sqrt{\sum_{j=1}^{m} (Y_j^+ - y_{ij})^2} \tag{13}$$

$$d_i^- = \sqrt{\sum_{j=1}^{m} (Y_j^- - y_{ij})^2} \tag{14}$$

where d_i^+ and d_i^- are the Euclidean distance between each index value and the positive/ negative ideal solution.

The pseudo-code for obtaining the best cache node is shown in Algorithm 1.

Algorithm 1 Obtaining the Best Cache Node Algorithm

Input: Cache hit node v_{hit} and a request content C
Output: v_c

```
if v_hit is not an edge node or a content provider and content
  C is a popular content in node v_hit then
    Use entropy weighting method and TOPSIS to calculate
    cache node v_c;
else if v_hit is a content provider then
    v_c ← the downstream neighbor node of the content
    provider on the delivery path;
return v_c;
```

B. TWO CACHE REPLACEMENT ALGORITHMS

In NDN, wire-speed forwarding makes routers unable to support complex replacement strategies [36]. Due to the simplicity and efficiency of LRU, most cache strategies use LRU as the default cache replacement strategy. In fact, LRU can store recent popular content in a single routing node, but it cannot reduce cache redundancy in the whole network [37]. In order to further improve cache utilization, two lightweight cache replacement algorithms are proposed including an active cache replacement algorithm and a passive cache replacement algorithm.

1) ACTIVE CACHE REPLACEMENT ALGORITHM

The active cache replacement algorithm aims to reduce the cache redundancy on the request path as much as possible, thereby reducing the cache redundancy of the whole network gradually. To implement the active cache replacement strategy, an *Eviction* field for evicting content objects is added to the interest packet. The *Eviction* field of the interest packet consists of two parts: the flag bit F with a default value of 0 and the eviction list (EL). If $F = 1$, it means that this request event not only requests the content object but also evicts the redundant content in the node on the request path according to the information in EL of the interest packet during the forwarding process. To reduce the extra pressure on route nodes, the active cache replacement strategy is executed when the number of interest packets received by each edge node reaches an integer multiple of the threshold T. The threshold T is denoted as the product of the average number of hops and the cache capacity (i.e. the maximum number of contents that can be cached by a single routing node.) of a single route node. According to the architecture of the content caching mechanism [38], the average number of hops is set to 7.

The whole eviction process is divided into two steps: updating the EL and evicting the content object. Step 1: updating the EL. If the cache hit node is not an edge node and $F = 1$, the EL will obtain the information of the CS in the edge node for initialization. When the interest packet

is forwarded to the next route node, the intersection of the information in the current node's CS and the original EL will be used to update the original EL. Step 2: evicting the content object. Assuming that the interest packet is forwarded to node v_i. If node v_i has only one downlink face and one uplink face (i.e. the degree of the node v_i is 2), it means that node v_i only serves the consumers on the connection current path. To improve cache utilization, the active replacement strategy evicts the redundant content in node v_i according to the information in EL. Particularly, if the degree of the node v_i is greater than 2, it means that node v_i may serve more consumers on different paths. In this case, it is difficult to judge whether the contents recorded in the ET are redundant for consumers on the node v_i downlink. Thus, the active replacement strategy only evicts the unpopular content objects in the intersection of node v_i and ET to maximize cache performance. The pseudo-code for the active cache replacement strategy is shown in Algorithm 2.

Algorithm 2 Active Cache Replacement Algorithm

Input: Number of received requests N_{E_j} for edge node E_j
Output: None
1 **if** N_{E_j} % (7 * B) == 0 **then**
2 $F = 1$; /*F is the flag bit in the Nth interest C arriving at edge node E_j.*/
3 **if** $F = 1$ **and** interest C does not cache hit at edge node E_j **then**
4 Copy the content name in CS of edge node E_j to EL of interest C;
5 **for** node v_i on the request path **do**
6 interest C does not cache hit at node v_i;
7 Update the El of interest C by the union of CS of node v_i and the original El;
8 **if** $D_{v_i} == 2$ **then**
9 **for** content k in the CS of node v_i **do**
10 **if** v_i in the El of interest C **then**
11 delete content k from the CS of node v_i;
12 **else if** $D_{v_i} > 2$ **then**
13 **for** content k in the CS of node v_i **do**
14 **if** v_i in the El of interest C **and** $r_{v_i}^k >= 80\%$ **then** /*The ranking range of cache hit count of content k at node v_i is greater than 80% */
15 delete content k from the CS of node v_i;
16 **return** None;

2) PASSIVE CACHE REPLACEMENT ALGORITHM

The passive cache replacement algorithm aims to improve the utilization of single content as much as possible, divided into two steps: the placement of the new content objects and the migration of discarded content objects.

In our scheme, the discarding rules of content objects are determined by the cache hit count of each content object in the current node. When the node v_i decides to cache a new content object, if the node v_i still has free cache space, then v_i will cache the new content object until the node v_i is full. Conversely, if the node v_i is full, the object of cached content, which has the lowest cache hit count, will be the candidate to be discarded from the node v_i. Moreover, if there are multiple content objects with the same cache hit count at node v_i, then one of them will be randomly selected as a candidate for discarding. To avoid that the storage time of a new content object cached in the node is too long or too short, our scheme places the new content object in the middle position of the cache node (i.e. the initial value of the cache hit count of the new content object is set to the median of the cache hit count of all content objects at the current node.). The impact of the placement of the new content object in the node on the cache performance will be analyzed in the simulation experiment in Section V.

Algorithm 3 Passive Cache Replacement Algorithm and Cache Mechanism

Input: Cache node v_c and cache object content C
Output: None
1 **if** v_c is not full **then**
2 Cache content C;
3 $N_{v_c}^C = N_{v_c}^{median}$;
4 **else if** v_c is full **and** v_c is an edge node or a normal node **then**
5 Migrate content L_{v_c} to node v_c^u;
6 Cache content C;
7 $N_{v_c}^C = N_{v_c}^{median}$;
8 **if** $N_{v_c}^{L_{v_c}} > N_{v_u}^{L_{v_c^u}}$ **then**
9 Cache content L_{v_c};
10 $N_{v_u}^{L_{v_c}} = N_{v_c}^{L_{v_c}}$;
11 **else if** v_c is full **and** v_c is the downstream neighbor node of the content provider **then**
12 Migrate content L_{v_c} to node v_c^d;
13 Cache content C;
14 $N_{v_c}^C = N_{v_c}^{median}$;
15 **if** $N_{v_c}^{L_{v_c}} > N_{v_u}^{L_{v_c^d}}$ **then**
16 Cache content L_{v_c};
17 $N_{v_u}^{L_{v_c}} = N_{v_c}^{L_{v_c}}$;
18 **return** None;

Since the proposed scheme only considers the local content popularity (i.e. the cache hit count for each content in the current node) in each node, the discarded content objects in the current node are likely to be popular in other nodes. To maximize the utilization of discarded content objects, the discarded content objects will migrate one hop up or down the request path according to different node types to improve cache performance. The proposed scheme divides nodes into three types: edge nodes, normal nodes, and the downstream neighbor node of the content provider. The migration rules for discarded content objects from different types of nodes are as follows: (i) When a discarded content object L_{v_c} comes from an edge node or a normal node, the content L_{v_c} will be migrated to an upstream node v_c^u one hop away from the edge node or the normal node along the delivery path. If the

node v_c^u still has free cache space, then the content L_{v_c} will be cached. Conversely, if the node v_c^u is full and the cache hit count of the content L_{v_c} is greater than the lowest cache hit count at the node v_c^u, the content L_{v_c} will be cached and the content with the lowest cache hit count at node v_c^u will be deleted. The cache hit count of the content L_{v_c} at the node v_c^u remains unchanged. (*ii*) When a discarded content object L_{v_c} comes from the downstream neighbor node of the content provider, the content L_{v_c} will be migrated to a downstream node v_c^d one hop away from the node along the delivery path. The caching mechanism of content L_{v_c} is the same as that in the migration rule (*i*). The pseudo-code for passive cache replacement strategy and cache mechanism strategy is shown in Algorithm 3.

V. PERFORMANCE EVALUATION

To evaluate the performance of our scheme, we performed a simulation in Tiscali-3257 network topology by using Icarus simulator [39]. The Tiscali-3257 topology has 44 provider nodes, 36 consumer nodes, and 160 router nodes. In the simulation experiment, all the caching strategies for comparison use the least replacement strategy (LRU). The cache capacity of all router nodes is the same, and the cache to content objects population ratio is S (i.e. the total capacity of network caches as a fraction of the total content population). In particular, only router nodes on request paths have cache capabilities. The link delay between the router node and the provider node is 34ms, and the link delay between the remaining nodes is 2ms [40], [41]. The process of consumers requesting content follows the Poisson distribution. The content popularity is modeled by Zipf distribution. The default value of the Zipf parameter α is 0.8, and it is varied between 0.7 and 1.1. Due to the high complexity of some caching strategies, such as literature [42], it is only suitable for scenarios with a small content catalog. To prove that the proposed scheme can be implemented smoothly with a large content catalog size, the total number of content objects in the network is set to 100,000. The cache warms up 50,000 requests and subsequent 250,000 requests are used for performance evaluation. The experimental parameters are shown in Table 2. To reduce experimental errors, five simulation experiments were carried out in total, and the final result is the average of these five simulation experiments results.

TABLE 2. Main experimental parameters.

Parameters	Values
Network topology	Tiscali-3257 topology
Content distribution model	Zipf distribution, $\alpha \epsilon [0.7,\ 1.1]$
Request rate	12 requests per second
Network caching capacity (S)	S$\in$ [0.05, 0.1, 0.15, 0.2, 0.25]
Link delay	2ms (between the router nodes) 34ms (between router node and provider node)
Number of content objects	100,000 messages
Number of warm up requests	50,000 messages
Number of measured requests	250,000 messages
Number of content objects	100,000 messages
Number of experiments	5

A. CACHE HIT RATIO

The cache hit rate metric is part of the request satisfied by the router nodes instead of the providers, thereby balancing the content requests among the available cache resources. If consumers can obtain the requested content in the router nodes, the pressure on the providers will be greatly reduced. Therefore, the cache hit ratio is one of the important metrics to evaluate the performance of NDN. The cache hit rate is calculated by the following formula:

$$CHR = \frac{\sum_{k=1}^{K} N_k}{NUM} \tag{15}$$

where K is the number of consumers, and N_k is the cache hit count of each consumer. NUM is the number of requests from all consumers.

As shown in Fig. 3 (a), the proposed scheme performs better in comparison to the five caching strategies in all the simulation experiments consistently where the Zipf parameter $\alpha = 0.8$ and the cache capacity parameter S varies from 0.05 to 0.25. The cache hit ratio of the proposed scheme is 26.6% when parameter S equals 0.05, which 2.1% higher than the second-best caching strategy (i.e. the cache hit ratio of our previous scheme CBCP is 24.5%) and 4.4% higher than the third-best caching strategy (i.e. the cache hit ratio of LCD is 22.2%). When parameter S equals 0.25, in comparison to CBCP and LCD, the cache hit rate of the proposed scheme increases by 1.2% and 6.3%, respectively. The average cache hit rate of the proposed scheme is 35.5%, which is 1.2% higher than CBCP (34.3%), 5.4% higher than LCD (30.1%), 10.6% higher than CL4M (24.9%), 11.8% higher than LCE (23.7%), and 13.9% higher than ProbCache (21.6%) when the parameter S varies ranging between 0.05 and 0.25. As shown in Fig. 3 (b), when the Zipf parameter α is increased to 1.0, the cache hit rate of all cache strategies is significantly improved, but the performance ranking of the cache strategies remains unchanged.

Fig. 3 (c) shows the changes in the cache hit rate performance with parameter S = 0.15 and increasing Zipf parameter α. The cache hit rate of the six cache strategies has an obvious improvement with the increase of Zipf parameter α. The cache hit rate of the proposed scheme is up to 77.1%, which is 1.6% higher than CBCP (75.5%), 4.8% higher than LCD (72.3%) when Zipf parameter S equals 1.1. For each caching strategy, the performance gain obtained

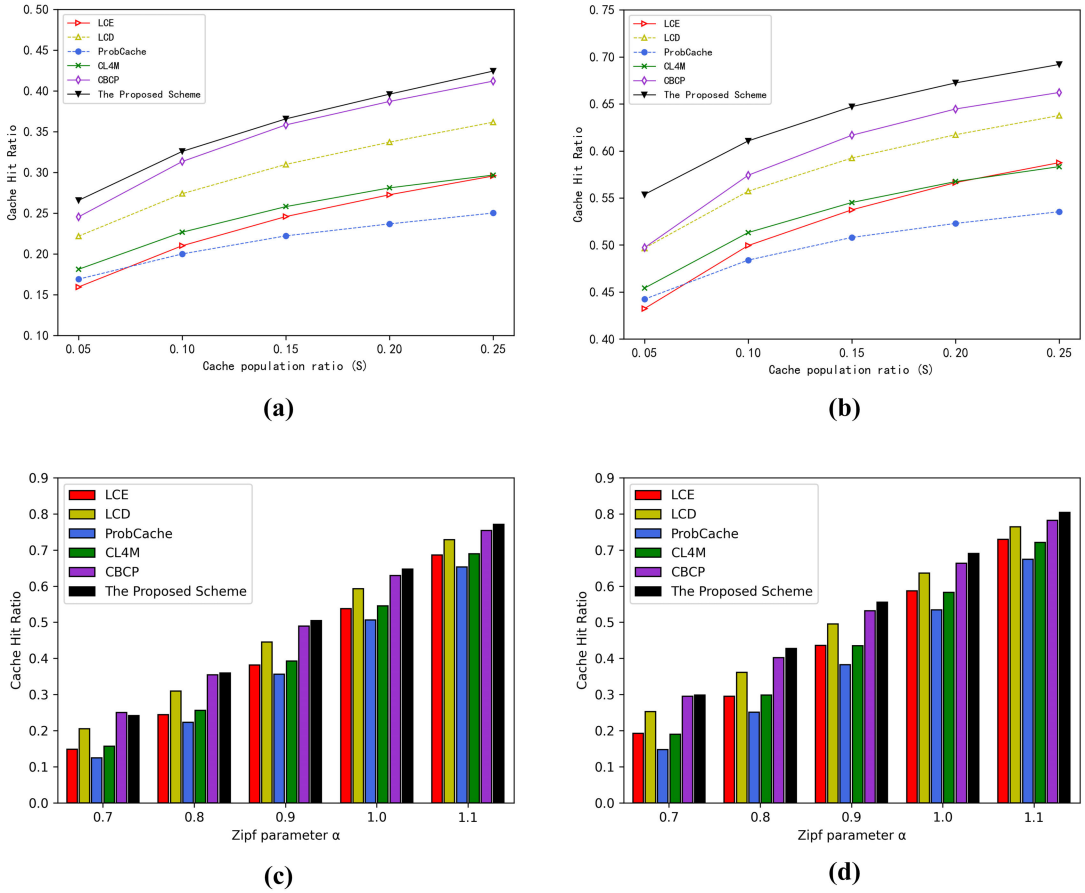


FIGURE 3. Cache hit ratio. (a) Different cache population ratio (S) ($\alpha = 0.8$). (b) Different cache population ratio (S) ($\alpha = 1.0$). (c) Different Zipf parameter α (S = 0.15). (d) Different Zipf parameter α (S = 0.25).

by changing Zipf parameter α is obviously greater than that by changing parameter S. The reason for this situation is that consumers are more inclined to request popular content with the increase of Zipf parameter α, which is a key factor affecting the cache hit rate. Fig. 3 (c) and Fig. 3 (d) show similar performance in terms of the cache hit ratio.

In summary, the proposed scheme can improve the cache hit rate in various situations. The main reason for the proposed scheme performs well is that it considers the content popularity in each node and the placement of popular content, thereby improving the utilization of the cache, especially the nodes with less pressure and edge nodes in the network. Different from CBCP which considers the global content popularity and the edge local content popularity, the proposed scheme focuses on the content popularity in each node to improve the performance of the entire network by optimizing the individual node.

B. LATENCY

Latency is an important performance to measure the network. The latency of the six caching strategies is shown in Fig. 4 (a) where the Zipf parameter $\alpha = 0.8$ and the cache capacity parameter S varies from 0.05 to 0.25. When parameter S equals 0.25, all caching strategies have the lowest latency, where the proposed scheme is 58.91 ms, which is 0.5% lower than CBCP (59.18 ms), 6.7% lower than LCD (63.15 ms), 13.3% lower than CL4M (67.95 ms), 14.4% lower than LCE (68.85 ms), and 17.4% lower than ProbCache (71.35 ms). In terms of average latency, in comparison to CBCP (64.65 ms) and LCD (67.90 ms), the proposed scheme is 64.07 ms, which is a reduction of 0.9% and 5.6%, respectively. In general, when the parameter S increases, the latency is naturally decreased because more content objects could be cached in the network. Moreover, the proposed scheme also fully considers the *HOP* index of the node when calculating the cache node, so the proposed scheme has lower latency. As shown in Fig. 4 (b), when the Zipf parameter α is increased to 1.0, the average latency of the proposed scheme is 39.63 ms, which still performs best.

In addition to the cache capacity parameter S, the Zipf parameter α also has a huge impact on the latency. Fig. 4 (c) shows the changes in the latency performance of each caching strategy with parameter S = 0.15 and increasing Zipf parameter α. The performance trend of latency is the same as the cache hit rate. With the increase of the Zipf parameter α, the latency performance of the six caching strategies has been

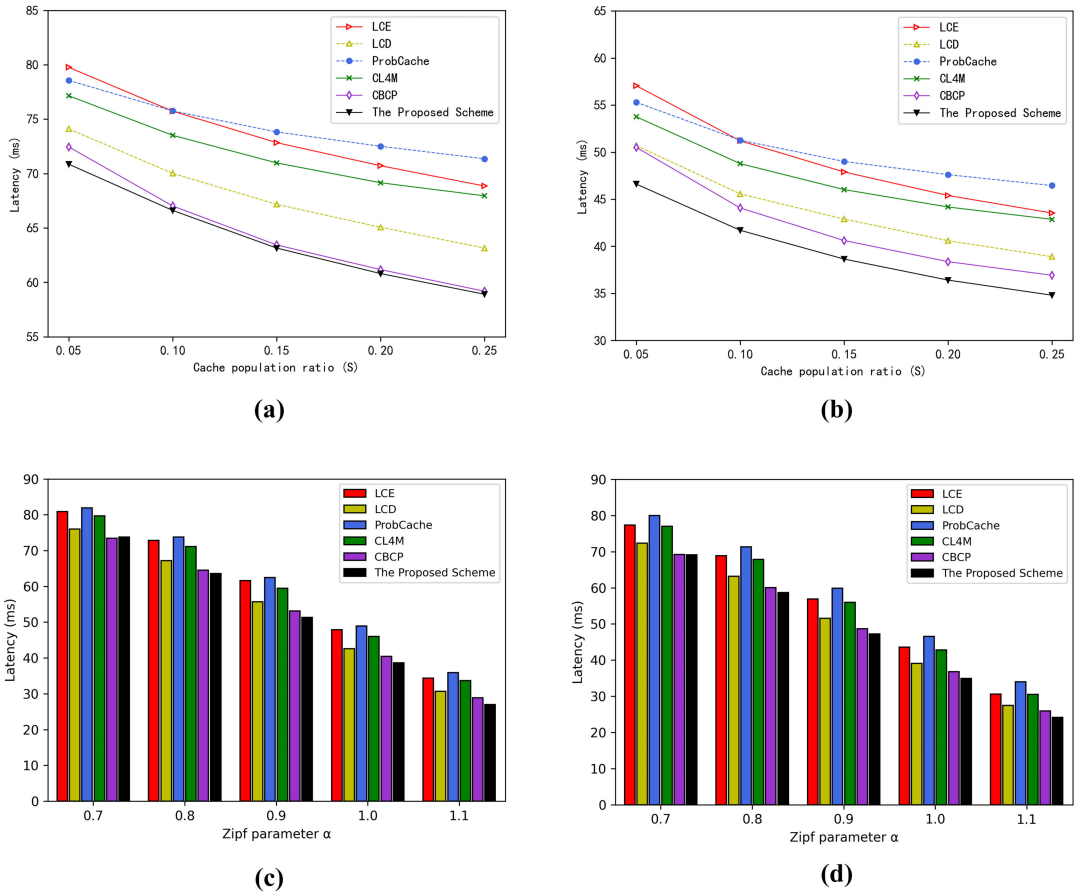


FIGURE 4. Latency. (a) Different cache population ratio (S) ($\alpha = 0.8$). (b) Different cache population ratio (S) ($\alpha = 1.0$). (c) Different Zipf parameter α (S = 0.15). (d) Different Zipf parameter α (S = 0.25).

significantly improved. The latency of the proposed scheme has the best performance in each value of Zipf parameter α. Specifically, the average latency of the proposed scheme is 50.87 ms, which is 2.3% lower than CBCP (52.09 ms), and 6.6% lower than LCD (54.44 ms) when the Zipf parameter α varies ranging between 0.7 and 1.1. As shown in Fig. 4 (d), when the parameter S is increased to 0.25, the overall change trend of all caching strategies remains basically unchanged.

C. LINK LOAD

The link load is the average traffic load of all links in the network. In Icarus, the link load of each link can be calculated by measuring the number of interest packets and data packets per unit time.

Fig. 5 (a) shows the link load performance against a varying cache capacity parameter S for a value of Zipf parameter α equal to 0.8. When parameter S equals 0.05, the proposed scheme performs significantly better compared to the other five caching strategies. However, when the parameter s varies from 0.1 to 0.25, the link load performance of the proposed scheme is not optimal. The average link load of the proposed scheme all evaluated scenarios is 261.16 bytes, which 1% lower than LCD (263.67 bytes), 2.4% lower than CL4M (267.66 bytes), 4.4% lower than ProbCache (273.06 bytes), 7.1% lower than LCE (281.03 bytes), and only 0.8% higher than CBCP (259.03 bytes). Generally, the link load performance of the proposed scheme is better than the LCD, LCE, CL4M, and ProbCache. Link load increment happens due to the fact that the proposed scheme does not consider global content popularity, in comparison to CBCP. Nevertheless, the proposed scheme can reduce the link load to a certain extent when the value of cache capacity parameter S is small. As shown in Fig. 5 (b), when the Zipf parameter α is increased to 1.0, the link load performance of all caching strategies is obviously improved. It is worth noting that the link load performance of the proposed scheme is basically the same as that of CBCP.

The influence of Zipf parameter α on link load is shown in Fig. 5 (c), which shows the changes in the link load performance with parameter S = 0.15 and increasing Zipf parameter α. The link load of the proposed scheme has the best performance when the Zipf parameter α varies ranging between 0.7 and 1.1. It is worth noting that the performance of the proposed scheme is almost the same as that of CBCP. This happens because the increase of the Zipf parameter α makes the difference between global popular content and popular

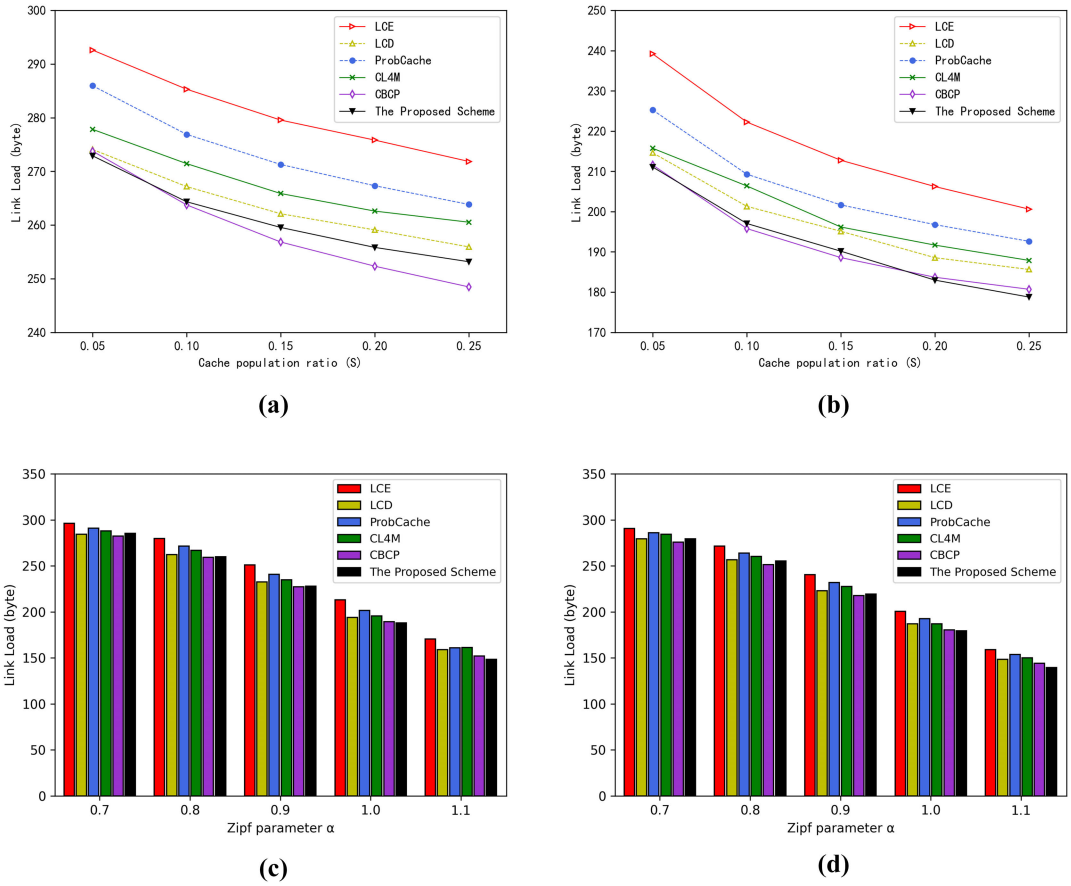


FIGURE 5. Link Load. (a) Different cache population ratio (S) ($\alpha = 0.8$). (b) Different cache population ratio (S) ($\alpha = 1.0$). (c) Different Zipf parameter α (S = 0.15). (d) Different Zipf parameter α (S = 0.25).

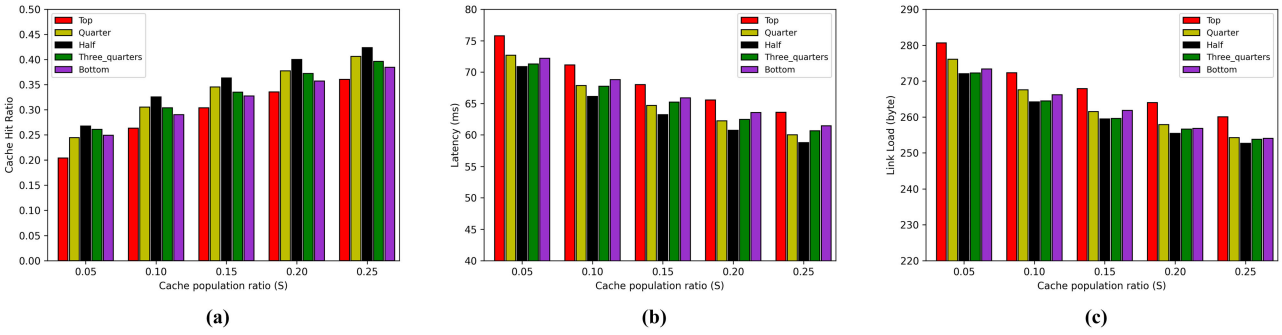


FIGURE 6. Different content placement positions in nodes on cache performance. (a) Cache hit ratio. (b) Latency. (c) Link load.

content within a single node smaller and smaller. When the parameter S is increased to 0.25, as shown in Fig. 5 (d), the overall change trend of all caching strategies is similar.

D. ANALYSIS OF THE CONTENT PLACEMENT POSITION IN THE NODE

In the proposed scheme, the placement position of a cache object in the node (i.e. the initial value of cache hit count of a cache object) will directly affect the cache performance. Thus, we analyzed the impact of the five placement positions within the node on the cache hit rate, latency, and link load. The five placement positions are the top, quarter, middle, three-quarter, and bottom positions within the node. The initial value of cache hit count corresponding to each cache location can be obtained by descending order of cache hit count of all contents in the node.

Figs. 6(a), 6(b), and 6(c) show the influence of different content placement positions in nodes on cache performance. It can be seen from the figures that the performance of the cache hit rate, latency, and link load is the worst when the

cache object is placed at the top. This happens because, the initial value of the cache hit count of the cache object is set very largely, which makes it difficult for this cache object to be discarded, even if the cache object gradually becomes unpopular content after a period of time. On the contrary, when the cache object is placed at the bottom, the initial value of the cache hit count of the cache object is set very small, which makes it easy for this cache object to be discarded quickly, resulting in the cache object not being fully utilized. Thus, when the cache object is placed at the bottom, the performance of the cache hit rate, latency, and link load is not very good either. Furthermore, there are similar problems when the cache object is placed at one quarter or three-quarter, so it is not the best choice to place the cache object in these positions. Thus, to alleviate the above problems to a certain extent, the proposed scheme chooses the middle position of the node to place the cache object to maximize the cache performance.

VI. CONCLUSION

One of the important features of the NDN architecture network is that it has an in-network caching function, allowing content objects to be cached in the router nodes for a period of time to satisfy the subsequent requests of consumers. The current study proposes a new caching strategy to manage the distribution of content in the network to improve the performance of the NDN system. In the scheme, the real-time status of nodes and the content popularity within a single node are considered comprehensively, and the request process of consumers is modeled by the TOPSIS methodology with information entropy weighting methodology to obtain the most suitable cache node for the cache object. In addition, we also propose a novel cache replacement algorithm to replace LRU commonly used in NDN. The cache replacement algorithm is composed of two parts: an active cache replacement algorithm and a passive cache replacement algorithm. The active cache replacement algorithm aims to periodically clear the duplicate content objects on the transmission path to reduce cache redundancy, while the goal of the passive cache replacement algorithm is to improve the utilization of data packets in the node. To evaluate the proposed caching strategy, a simulation environment was created using the Icarus simulator in the Tiscali-3257 network topology. The results show that the proposed scheme has the best performance in the cache hit rate and latency in comparison to the other five caching strategies, and only the link load performance is slightly lower than our previous scheme CBCP.

For future research, we will further analyze the relationship between nodes and content objects in the mass and improve the caching strategy to reduce link load.

REFERENCES

[1] Cisco. (Mar. 9, 2020). *Cisco Annual Internet Report (2018-2023)*. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[2] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi, "A survey on content-oriented networking for efficient content delivery," *IEEE Commun. Mag.*, vol. 49, no. 3, pp. 121–127, Mar. 2011.

[3] R. Steinmetz and K. Wehrle, "Peer-to-peer-networking &-computing," *Informatik Spektrum*, vol. 27, no. 1, pp. 51–54, Feb. 2004.

[4] S. Akhshabi and C. Dovrolis, "The evolution of layered protocol stacks leads to an hourglass-shaped architecture," in *Proc. ACM SIGCOMM Conf.*, vol. 41, no. 4, Feb. 2011, pp. 206–217.

[5] A. V. Vasilakos, Z. Li, G. Simon, and W. You, "Information centric network: Research challenges and opportunities," *J. Netw. Comput. Appl.*, vol. 52, pp. 1–10, Jun. 2015.

[6] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 2, pp. 1024–1049, 2nd Quart., 2014.

[7] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, New York, NY, USA, Aug. 2007, pp. 181–192.

[8] C. Dannewitz, J. Golic, B. Ohlman, and B. Ahlgren, "Secure naming for a network of information," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM)*, San Diego, CA, USA, Mar. 2010, pp. 1–6.

[9] P. Jokela, A. Zahemszky, C. E. Rothenberg, S. Arianfar, and P. Nikander, "LIPSIN: Line speed publish/subscribe inter-networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 195–206, Aug. 2009.

[10] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. 5th Int. Conf. Emerg. Netw. Exp. Technol.*, Dec. 2009, pp. 1–12.

[11] P. Agyapong and M. Sirbu, "Economic incentives in information-centric networking: Implications for protocol design and public policy," *IEEE Commun. Mag.*, vol. 50, no. 12, pp. 18–26, Dec. 2012.

[12] M. Amadeo, C. Campolo, G. Ruggeri, G. Lia, and A. Molinaro, "Caching transient contents in vehicular named data networking: A performance analysis," *Sensors*, vol. 20, no. 7, pp. 1–17, Apr. 2020.

[13] N. L. M. van Adrichem and F. A. Kuipers, "NDNFlow: Software-defined named data networking," in *Proc. 1st IEEE Conf. Netw. Softwarization (NetSoft)*, London, U.K., Apr. 2015, pp. 13–17.

[14] M. Wang, J. Wu, G. Li, J. Li, and Q. Li, "Fog computing based content-aware taxonomy for caching optimization in information-centric networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Atlanta, GA, USA, May 2017, pp. 474–475.

[15] T. Jin, X. Zhang, Y. Liu, and K. Lei, "BlockNDN: A bitcoin blockchain decentralized system over named data networking," in *Proc. 9th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Milan, Italy, Jul. 2017, pp. 75–80.

[16] O. Ascigil, S. Reñé, G. Xylomenos, I. Psaras, and G. Pavlou, "A keyword-based ICN-IoT platform," in *Proc. 4th ACM Conf. Inf.-Centric Netw.*, Sep. 2017, pp. 22–28.

[17] M. Naeem, R. Ali, B.-S. Kim, S. Nor, and S. Hassan, "A periodic caching strategy solution for the smart city in information-centric Internet of Things," *Sustainability*, vol. 10, no. 7, p. 2576, Jul. 2018.

[18] T. Zhang, S. Shan, X. Xu, and Y. Liu, "Survey on caching techniques of information centric networking," *J. Beijing Univ. Posts Telecommun.*, vol. 39, no. 3, pp. 1–15, Mar. 2016.

[19] Y. Gui and Y. Chen, "A cache placement strategy based on compound popularity in named data networking," *IEEE Access*, vol. 8, pp. 196002–196012, Oct. 2020.

[20] S. Vural, P. Navaratnam, N. Wang, C. Wang, L. Dong, and R. Tafazolli, "In-network caching of Internet-of-Things data," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Sydney, NSW, Australia, Jun. 2014, pp. 3185–3190.

[21] B. Nour, K. Sharif, F. Li, S. Biswas, H. Moungla, M. Guizani, and Y. Wang, "A survey of Internet of Things communication using ICN: A use case perspective," *Comput. Commun.*, vols. 142–143, pp. 95–123, Jun. 2019.

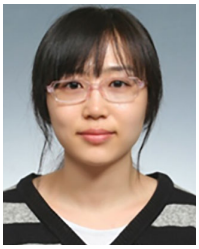
[22] N. Laoutaiis, S. Syntila, and L. Stavrakakis, "Meta algorithms for hierarchical Web caches," in *Proc. IEEE Int. Conf. Perform., Comput., Commun.*, Apr. 2004, pp. 445–452.

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

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

[25] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proc. 2nd Ed. ICN Workshop Inf.-Centric Netw. (ICN)*, Aug. 2012, pp. 55–60.

[26] Q. N. Nguyen, J. Liu, Z. Pan, I. Benkacem, T. Tsuda, T. Taleb, S. Shimamoto, and T. Sato, "PPCS: A progressive popularity-aware caching scheme for edge-based cache redundancy avoidance in information-centric networks," *Sensors*, vol. 19, no. 3, pp. 1–18, Feb. 2019.
[27] H. Wu, J. Li, and J. Zhi, "Probability-based heuristic content placement method for ICN caching," *J. Commun.*, vol. 37, no. 5, pp. 62–72, 2016.
[28] G. Zhang, Y. Hu, W. Huang, B. Wang, and L. Cao, "Coordinated caching scheme based on popular content awareness and tracking," *J. Commun.*, vol. 38, no. 2, pp. 132–142, Feb. 2017.
[29] L. Saino, I. Psaras, and G. Pavlou, "Hash-routing schemes for information centric networking," in *Proc. 3rd ACM SIGCOMM Workshop Inf.-Centric Netw. (ICN)*, Aug. 2013, pp. 27–32.
[30] Y. Yang, T. Song, and B. Zhang, "OpenCache: A lightweight regional cache collaboration approach in hierarchical-named ICN," *Comput. Commun.*, vol. 144, pp. 89–99, Aug. 2019.
[31] Z. Zhang, C.-H. Lung, M. St-Hilaire, and I. Lambadaris, "Smart proactive caching: Empower the video delivery for autonomous vehicles in ICN-based networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7955–7965, Jul. 2020.
[32] Y. Hua, L. Guan, and K. G. Kyriakopoulos, "A fog caching scheme enabled by ICN for IoT environments," *Future Gener. Comput. Syst.*, vol. 111, pp. 82–95, Oct. 2020.
[33] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," *Comput. Netw.*, vol. 57, no. 16, pp. 3128–3141, Nov. 2013.
[34] R. Jmal and L. C. Fourati, "An OpenFlow architecture for managing content-centric-network (OFAM-CCN) based on popularity caching strategy," *Comput. Standards Interfaces*, vol. 51, pp. 22–29, Mar. 2017.
[35] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *Proc. IEEE Conf. Comput. Commun. 18th Annu. Joint Conf. IEEE Comput. Commun. Soc.*, New York, NY, USA, Mar. 1999, pp. 126–134.
[36] J. Duan, Y. Xing, and G. Zhao, "Survey for caching technologies in information centric networking," *Comput. Eng. Appl.*, vol. 54, no. 2, pp. 1–10, 2018.
[37] Y. Wang, K. Lee, B. Venkataraman, R. L. Shamanna, I. Rhee, and S. Yang, "Advertising cached contents in the control plane: Necessity and feasibility," in *Proc. IEEE INFOCOM Workshops*, Orlando, FL, USA, Mar. 2012, pp. 286–291.
[38] K. Xu, M. Zhu, N. Wang, S. Lin, H. Wang, and T. Li, "The 2ACT model-based evaluation for in-network caching mechanism," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Split, Croatia, Jul. 2013, pp. 636–641.
[39] L. Saino, I. Psaras, and G. Pavlou, "Icarus: A caching simulator for information centric networking (ICN)," in *Proc. 7th Int. Conf. Simul. Tools Techn.*, Mar. 2014, pp. 66–75.
[40] B. Zhang, T. S. E. Ng, A. Nandi, R. H. Riedi, P. Druschel, and G. Wang, "Measurement based analysis, modeling, and synthesis of the interne delay space," in *Proc. 6th ACM SIGCOMM Conf. Internet Meas.*, Oct. 2006, pp. 85–98.
[41] J. Rajahalme, M. Särelä, K. Visala, and J. Riihijärvi, "On name-based inter-domain routing," *Comput. Netw.*, vol. 55, no. 4, pp. 975–986, Mar. 2011.
[42] X. Xu, C. Feng, S. Shan, T. Zhang, and J. Loo, "Proactive edge caching in content-centric networks with massive dynamic content requests," *IEEE Access*, vol. 8, pp. 59906–59921, Mar. 2020.



YIQI GUI received the B.S. degree in computer science from Beihua University, China, in 2004, and the M.S. and Ph.D. degrees in computer information and communication engineering from Kangwon National University, South Korea, in 2007 and 2012, respectively.

From 2007 to 2009, she was a Research Assistant with the Database and Multimedia Laboratory. Since 2012, she has been an Assistant Professor with the Computer Science Department, Yangzhou University, China. She was a Visiting Scholar with Kangwon National University, in 2017. Her research interests include peer to peer multimedia systems, information-centric networking (ICN) and future networks, and network security. She is currently a member of CCF and also a reviewer of several SCI-indexed journals and international conferences.

YONGKANG CHEN received the B.S. degree in computer science from Nantong University, China, in 2016.

He is currently pursuing the M.S. degree with Yangzhou University, China. His research interests include datamining for multimedia systems and network caching technology for information-centric networking (ICN).