Received August 18, 2019, accepted September 26, 2019, date of publication October 8, 2019, date of current version November 6, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946184

EHCP: An Efficient Hybrid Content Placement Strategy in Named Data Network Caching

YAHUI MENG[1], MUHAMMAD ALI NAEEM[1], RASHID ALI[2], AND BYUNG-SEO KIM[3], (Senior Member, IEEE)

[1]School of Science, Guangdong University of Petrochemical Technology, Maoming 525000, China
[2]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 712-749, South Korea
[3]Department of Software and Communications Engineering, Hongik University, Sejong 30016, South Korea

Corresponding authors: Muhammad Ali Naeem (malinaeem7@gmail.com) and Byung-Seo Kim (jsnbs@hongik.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) through the Korean Government under Grant 2018R1A2B6002399.

ABSTRACT The abilities of in-network caching in Named Data Networking (NDN) agrees to enable effective data dissemination throughout the globe without requiring the end-to-end communication infrastructure. The goal of this study is to develop an efficient content placement strategy for the NDN caching module to achieve the enhanced performance for the NDN network. In this work, optimizations of the problems in NDN-based caching strategies are formulated and to solve these problems, a new caching strategy is designed Named as Efficient Hybrid Content Placement (EHCP). The aim of the EHPC strategy is to reduce the multiple replications of homogeneous contents at numerous locations during data dissemination and to increase the diverse number of contents along the data delivery path. In addition, the effect of the cache-hit ratio, hop-decrement, and content redundancy is shown by comparing the proposed caching strategy with other state-of-the-art caching strategies in a simulation environment. The consequences show that the proposed strategy delivers an enhanced number of heterogeneous contents in terms of improving content diversity ratio along with the higher cache-hit ratio. Moreover, with different cache sizes, EHCP performs better in terms of content redundancy and hop-decrement as compared to the existing state-of-the-art strategies.

INDEX TERMS Named data networking (NDN), hop-based probabilistic caching (HPC), video on demand (VoD), user generated content (UGC).

I. INTRODUCTION

The incredible growth of today's Internet traffic and frequent transmissions of the same contents over the Internet brings several issues and challenges, for instance, low bandwidth utilization, high server load, increased response time, and unnecessary usage of resources. In the future, IP-based architectures will be insufficient to solve these challenges due to end-to-end network infrastructure [1]. Usage of the Internet is increasing day by day, and the current resources are insufficient to handle the huge amount of requested contents [2]. Therefore, the future Internet needs to implement an optimal approach to tackle these increasing difficulties. For this purpose, the present Internet needs to enhance its architectural design in the future according to the expected requirements [3]. Recently an emerging Internet architecture such as Named Data Networking (NDN) is established that has the ability to provide a better solution to resolve such expected issues and challenges by implementing the in-network cache/content placement, which is the essential and unavoidable module of NDN architectures. In NDN, cache is temporary storage that is used to place contents (information) to speed up the communication process [4]. It is used to minimize the average cost (time and energy) and improve the content retrieval process by temporarily storing a copy of frequently requested data. Web caching [5] is a famous example of temporary storage in which, the web documents (such as HTML pages images) are cached to reduce the bandwidth consumption and server load. In-network Cache makes NDN different from the present IP-based Internet architecture. In this module, the transmitted content is cached near the user, which decreases the response time and resource utilization for subsequent users' requests. In addition, caching of content within the network nodes improves the data transmission

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Khalil Afzal.

services [6]. NDN caching module has on-path caching infrastructure [7]. Moreover, caching is managed by content placement strategies and content replacement policies. In on-path caching, the disseminated content is cached at intermediate network nodes during transmission of content from provider to the user. For the subsequent requests, in the on-path caching approach, a node immediately responds to the user's request with a locally cached content towards the user followed by the same path used by the request. This approach reduces the computation as well as communication overhead for subsequent requests [8]. A significant issue related to on-path is to make caching decisions by each node to improve the content delivery services. For example, highly requested content is placed at the node where it is requested more often. However, the information can be cached opportunistically on the data routing path because the subsequent requests for the same content follow the name resolution path, improve the possibility for a cache-hit ratio. Once the data routing and name resolution are coupled, and data is cached on the data routing path, a cache-hit occurs [8], [9].

Content placement is a process in which the user's requested content is cached along the data routing path in temporary storage (cache) located within the network nodes [2], [10]. It has a significant role to improve the overall network performance. For instance, the bandwidth and latency can be minimized by caching content at intermediate network nodes during accessing the desired contents [11]. The reason for this performance improvement is that all the subsequent requests are satisfied from that cached content. In addition, it reduces the server load and maximizes the availability of desired contents. Besides, it reduces the usage of network resources and minimizes network traffic [12]. The objective of these strategies is to forward the requested contents to the user and to cache a copy of transmitted content near the requesters to reduce the average cost for subsequent user's requests. Recently, several strategies for content placement have been proposed to manage the NDN in-network caching [9], such as Leave Copy Everywhere (LCE), [2] Leave Copy Down (LCD) [13], Move Copy Down (MCD) Probabilistic-based caching strategies [14], Unique Caching (UniCache) [2], and popularity-based caching strategies [15]. However, these strategies boost information replication and improve overall performance [16].

One of the main issues of these strategies is the capacity of cache, which is much smaller to accommodate all the incoming contents. Furthermore, efficient content distribution can be achieved by integrating a considerable amount of caches, which is not easy to provide a significant cache size. However, dynamic caching performance can be achieved by deploying an optimal content placement strategy. Therefore, researchers are trying to implement such an efficient approach that can produce efficient caching performance [17]. Therefore, several content placement strategies were proposed to handle the mentioned issues with high caching performance. However, it is not clear which caching strategy gives the optimal performance to solve all the presenting issues because the research for content placement in NDN caching is still in its early stage. In addition, the EHCP content placement strategy is developing in this research to resolve the caching issues of existing strategies. The EHCP handles the most recently downloaded contents as well as least recently download contents. Moreover, EHCP caches the most recently downloaded contents at central routers at data delivery path and least downloaded contents at edge routers.

TABLE 1. Acronyms.

Acronym	Notations
NDN	Named Data Networking
EHCP	Efficient Hybrid Content Placement
TSI	Time Since Inception
TSB	Time Since Birth
LCE	Leave Copy Everywhere
LCD	Leave Copy Down
CCAC	Cache Capacity Aware Cache
MFR	Most Frequently Requested
LRU	Least Replacement Policy
LPD	Leaf Popular Down
HPC	Hop-based Probabilistic Caching
UniCache	Unique Caching
MCD	Move Copy Down
PIT	Pending Interest Table
FIB	Forwarding Information Base

The rest of the paper is organized as follows. Section II provides related research work on content placement strategies. In section III, the problem statement is formulated by describing the issues of related works. Section IV presents our proposed EHCP content-placement approach. In section V, the evaluation of the proposed algorithm is done. In section VI, performance evaluation is discussed. Finally, section VII presents the conclusion.

II. RELATED WORK

In default NDN-based caching strategy, when a user request is received to download content, then a copy of the requested content is cached at all nodes available on the publisher-subscriber path. Thus the subsequent claims may be satisfied locally [18]. On-Path caching is a compatible module in which, caching strategies are used to cached requested contents near the users for the subsequent requests as well as to reduce the source and network load [9], [19]. The content caching mechanism depends on different criteria. For example, the caching decision can be accomplished either at the centrality of the network, probabilistic positions or by calculating the content popularity. Moreover, it can be done by adopting a mathematical equation or by measuring the distance from the source to the user. According to the naming granularity, the caching can be categorized into several types [12], [20]. For example, packet-level caching, chunk level caching, and object-level caching. According to the nature of the information, the caching models can be categorized as autonomous-model-based caching strategies, cooperative-model-based caching strategies, and centralized-model-based caching strategies [21].

LCE [22] caches the transmitted contents requested by a user, and the requested content is always searched at all on-path nodes. As the cache-hit occurs, the content is sent back to the appropriate user. According to LCE, a copy of the requested content is cached at all nodes along a publisher-subscriber path [23] and thus, the subsequent requests are satisfied by locally cached contents [16].

LCD [18] was developed to decrease the redundancy in LCE. LCD caches the requested contents only at the down-stream node along the data routing path. A path is created from subscriber to publisher for requested content. The publisher sends the required content back to the user and caches a copy of transmitted content only at the neighbor node, which is placed one level down from the publisher node [11]. On the other hand, probabilistic-based content placement strategies [21] have gained most of the researchers' interests because of their flexible selection of probabilistic positions for content caching. The probabilistic position depends on the nature of strategy algorithms, such as distance-based probabilistic position, random-based probabilistic position, fixed-based probabilistic position, or it can be dynamic-based probabilistic position such as ProbCache [21], ProbCache+ [24] and Hop-based Probabilistic Caching (HPC) [25].

In ProbCache, ProbCache+ and HCP [26], two primitives: Time Since Inception (TSI) and Time Since Birth (TSB) are created to measure the path length between user and publisher for the requested content. TSI is integrated inside the Interest header, and TSB is allocated to the content header [27]. The value of TSI is initialized and it assigns to each user-request. This value increases hope-by-hop as the request forwards to the publisher. TSB is attached inside the header of requested content as a hit materialized; its value is initialized to one. This value increases hop-by-hop as the content traverse towards the user. The TSI value is copied to the content header from the request header. However, this value remains unchanged because content is sent through data routing path [28].

TSI value indicates the path length covered by the user's request (from user to the publisher) and TSB represents the distance that is covered by content during its transmission from publisher to the user. Based on the ProbCache algorithm, the cache period, path computation, and hop distance is required to send a user request. Initially, at the time when a cache-hit is recorded, the requested data is sent out towards the user. Secondly, the requested content is cached probabilistically at all middle nodes for particular time span. Through PIT record, this operation is performed at the intermediate node. However, in continuous requests and network traffic, caching positions, cache time windows, and the path capacity of the network need to be obtained for each request.

Leaf Popular Down (LPD) [29] is a popularity-based content placement strategy, which caches the popular content at the downstream node and edge node. A path is created from the user to the publisher for a particular request to continue further communication. The publisher node adds an entry in its popularity table for each cache-hit. The content is selected as a popular content if the number of received requests for that content achieves threshold value (value to identify the popularity of the content). The popular content is sent back to the requested user, and a copy of the content is cached at a neighboring node, which is placed one level down from the publisher node, and one copy is cached at the edge router located one hop before the requested user.

Cache Capacity Aware Cache (CCAC) [18] proposes the composition of selective caching with a cache-aware routing algorithm. It was designed to reduce network load from the servers. CCAC observes recent cache consumption to estimate the available cache to accommodate new coming contents to be cached at a routing path. A node indicating the highest cache capacity has more chances to cache the upcoming content within its cache. The cache-aware routing forwards the user request to the suitable provider by using an extra feature Forwarding Information Base (FIB). The cache capacity is calculated using an inverse function to the amount of recently used cache. Therefore, the cache capacity value of the i^{th} node can be calculated as shown in the following,

$$CCV(i) = \frac{c}{L(i)} \times Cache_{size}(i) \tag{1}$$

where c represents the compensation value in the case of the huge caching load and L, is the distortion among the network nodes. Two fields as CCV_i and Network Distance Value is integrated with each Interest to find the available cache and distance between a consumer and a content provider. As the cache-hit is recorded, the content is forwarded to the consumer according to the following equation,

$$w_r = \frac{\log r}{logN_{total}} = \log_{N_{total}} r \tag{2}$$

where N_{total} shows the total number of contents and r represents the ranking of popular contents. The popularity of a content is measured by taking the sum of Interest generated for particular content. Therefore, a weight is associated with each popular content that helps the content to be cached with a higher possibility. The CCV_i is attached to the content as the provider makes a reply to the Interest, and the threshold is individually determined by combining the CCV_i and w_r at all nodes along the routing path.

$$CCV_{th} = CCV_{highest} \times w_r \tag{3}$$

where CCV_{th} represents a threshold, which is used to make content popular.

III. THE PROBLEM DESCRIPTION

NDN can minimize the expected flow of traffic in a network [25] by implementing the cache storage inside the network nodes (routers). However, the amount of transmitting data is much large as compared to the amount of cache storage available within network nodes. Due to restricted cache capacity, a number of problems are created [17], [30] such as; the content diversity, cache hit ratio, hop-decrement are reduced because of multiple caching operations are done by the similar contents. In addition, redundant content replications are increased. Effective caching performance demands an efficient cache management approach that can handle these issues with high performance. Subsequently, a number of NDN-based caching strategies have been proposed to manage cache inefficient manner. For example, LCE, LCD, Probabilistic cache, ProbCache, LPD, and CCAC. All these caching strategies increase the homogeneous content replications through caching the similar contents at multiple nodes that create a high level of content redundancy and reduces the content diversity. In these caching strategies, the similar contents cache at all the nodes along the data delivery path that minimizes the cache-hit ratio and increases the distance for the subsequent requests between the user and publisher. On the other hand, a large number of eviction operations are performed to accommodate the new transmitted content while the cache storage of the nodes along the data routing path is overflowed. Consequently, when a request for old content is received to the network, it needs to traverse several hops to download the required content from the remote publisher that increases the path length between the user and requested content. Thus, the hop-decrement is reduced and cache-hit ratio cannot retain its efficient level. ProbCache and CCAC strategies aim to distribute equal cache storage capacity among the connected nodes along data delivery path to improve the overall cache performance, but these strategies increase the inefficient usage of resource (cache) by caching the similar contents at multiple nodes along the unique data routing path. In addition, if the publisher-subscriber path is small then the frequent redundant operations are accomplished due to the large replications of similar contents.

ProbCache per se cannot accomplish its goals because it aims to reduce memory consumption by mitigating the content redundancy, but it seems fail to resolve these problems along the short-stretch data delivery path [31]. It does not provide the positioned distinction while the cache storage of a node is overflowed. As a result, the cache-hit is decreased because there is a possibility of caching contents far from the user that increases the distance between the cached content and the user. These strategies introduce various parameters to compute arbitrarily for each user request and requested content at all the nodes that increase the communication overhead. Furthermore, the ProbCache needs to update each user request and content in the form of TSI and TSB which increases the computational cost and reduces the cache-hit ratio. Table 2 illustrates the contributions and the challenges have been facing by the existing NDN-based caching strategies.

In fact, the efficient caching strategy needs to select the best possible position to store the transmitted content in such a fashion that can minimize the existing problems [13]. Thus, we propose a hybrid content selection-based caching strategy that will cache the most popular content as well as least popular content to solve the mentioned issues of related NDN-based caching strategies. The following section is presented the detailed explanation of EHCP caching strategy.

TABLE 2. Contributions and challenges in existing ndn-based caching strategies.

Caching Strategies	Contributions	Challenges
Leave Copy Everywhere (LCE)	• High cache hit ratio • Short stretch path	• High bandwidth • High redundancy ratio • Low diversity ratio • High memory usage
Leave Copy Down (LCD)	• High cache hit ratio • Short stretch path	• High redundancy ratio • Low diversity ratio • High memory consumption
Move Copy Down (MCD)	• Less usage of resources • High diversity ratio	• High delay • High stretch ratio • Less cache hit ratio
Unique Caching (UniCache)	• High diversity ratio • Low bandwidth	• Less cache hit ratio • Long delay • High stretch ratio • No position distinction
Priority Based Probabilistic Caching (PBPC)	• Low stretch ratio • Less delay	• High bandwidth • Less diversity ratio • High redundancy ratio • High usage of memory
Distance Probabilistic Position (DPP)	• High cache ratio • Less delay	• High usage of memory • High redundancy ratio • Low diversity ratio • High bandwidth
Random Probabilistic Caching (RPC)	• Less delay • High cache hit ratio • Less stretch ratio	• No position distinction • No distinction for popular content • Less diversity ratio
ProbCache	• High cache hit ratio • Low delay • Short stretch ratio	• No distinction for popular content • High redundancy ratio • Less diversity ratio • High bandwidth • High memory usage
ProbCache+	• High cache hit ratio • Low delay • Short stretch ratio	• No distinction for popular content • High redundancy ratio • Less diversity ratio • High bandwidth • High memory usage
Hop-based Probabilistic Caching (HPC)	• High cache hit ratio • Low delay • Short stretch ratio	• No distinction for popular content • High redundancy ratio • Less diversity ratio • High bandwidth • High memory usage
Leaf Popular Down (LPD)	• High cache hit ratio • Short stretch ratio	• Less diversity ratio • High redundancy ratio • High memory consumption

IV. PROPOSED CONTENT PLACEMENT STRATEGY

The objective of the NDN-based caching module is to reduce the arising issues of the present Internet paradigm. Therefore, NDN-based caching strategies try to handle these issues by implementing the cacheable nodes (routers) in the network. Cache storage is used to store the transmitted contents by deploying caching techniques [32]. These caching techniques can implement efficiently with the help of content placement strategies. A content placement strategy defines some rules to find the appropriate to be placed at intermediate

Algorithm 1 User Request

```
1. Get the parameters in the entire network
2.  (Source, Destination, content, Record)
3.  If the content is not obtaining edge_node, get content
    from centrality_node.
4.  Else
5.  if cache miss and no matched PIT entry
6.  then creates PIT entry and update the record
7. then forward the packet to the next node to the
      original provide
8. send requested DATA back to the requeste
```

Algorithm 2 Selection of Most Frequently Requested Content

```
def GetMFRContent(MFR_List, interest_List):
Icount = dict(interest, int);
 total_Contents=0
 for content in interest_List:
  if Icount.Contains(content):
      Icount[content] + = 1
    else:
      Icount.Add(content, 1)
      total_Contents + = 1
    RequestCount = len(interest_List)
     for content in Icount.keys(self):
     if Icount[content] > three_RequestCount:
        MFR_List.Add(content)
     else:
        break;
     i + =1
return MFR_List
```

Algorithm 3 Content Caching at Edge and Center

```
class EDGEANDCENTER(CacheManager)
 def _init_strategy(self)
    self.central = networkx.algorithms._centrality(self)
 def retrieve_from_caches(self, interest, path):
    content_found_caches = False
    for i in range(0, len(path)):
        p = path[i]
        if self.lookup_cache(p, interest):
            content_found_caches = True
             break
         else:
             pass
 #EDGE
 # In anycase, we put a copy in the first cacheable
element
    first = 0
    while first < len(path)-1
       first + = 1
     if first < i: #Only if the element is placed before the
original requester
       self.store_cache(path[first], interest)
   #CENTER
   #Cache in the node with the biggest centrality in the path
    max_v = -1
    node = None
    for i in range(0, len(path)):
       p = path[i]
       if self.central[p] > max_v:
           max_v = self.central[p]
           node = p
    self.store_cache(node, interest)
    return (content_found_caches, i)
```

location (node) between user and publisher to caches a copy of transmitted content to fulfill the requirements of subsequent requests. The appropriate position selection for transmitted content caching is a primary requirement to achieve efficient caching performance. A number of content placement strategies have been developed through implementing different criteria, such as probabilistic-based caching and popularity-based caching strategies [33]. In this study, a content placement technique named as Efficient Hybrid Content Placement (EHCP) strategy is proposed to enhance the overall NDN-based content caching mechanism.

EHCP is divided into two sections. In the first section, all the transmitted contents are cached at intermediate network nodes. In second section, the popular contents are cached at intermediate network nodes. The overall content caching mechanism is illustrated in Algorithm 3. In EHCP, all the requested contents are cached at edge nodes (without caring for the contents' popularities) to increase content availability. If a content is cached at edge nodes, the subsequent requests for that cached content will satisfy from the edge node. As a result, the cache-hit ratio will be reduced because most of the subsequent requests will be accomplished from nearly cached copy of that content. According to EHCP, the least recently downloaded content is cached at the edge node. However, the most recently downloaded content is cached at highest centrality node and at the edge node.

If a content is not selected as Most Frequently Requested (MFR), it cannot be cached at a centrality node that increases the chance to accommodate the MFR contents at the centrality node. The reason is that most of the subsequent requests for MFR contents will be accomplished from centrality node rather than to have a response from the remote node. In this way, the cache will be used in an efficient manner to enhance the cache-hit ratio.

In EHCP, each node is associated with a distinctive statistics table in which user requests are calculated to find the frequently requested content. To measure the requests frequency for particular content, the content name requests frequency count, and the threshold is required. Each node locally counts the number of received requests for each content and differentiate the contents into least popular and most popular according to their requests' frequencies. The threshold is the static value used to identify MFR content and it is managed by the strategy algorithm (Algorithm 2).

Algorithm 4 Node Selection (Centrality Node)

```
1. Centrality(SourcePosition, DesinationPosition,
   Content, LogRecord, topology)
2. Set central_node to
   networkx_Algorithms_Centrality(topology)
3. For i over Range from 0 to the length of the path
4. Set p to the path at index i
5. If any request found on any path within cache Then
6. Set Content_Cache
7. Terminate the Loop
8. Else
9. Continue loop
10. Set first to 0
11. While first is less than the length of the path And
    doesn't exist in caching_cababilities(path at index first)
    Increments first by 1
12. If first is less than i
13. Store path in the cache along with the request
14. Set max_v to -1
15. Set node to none
16. For i over Range from 0 to length of the path
17. Set p to the path at index i
18. If central_highest at index p is greater than max_v Then
19. Set max_v to central_highest at index p
20. Set node = p
21. Store node and request in the cache
22. Return content found within caches and the index i
```

As the requests for a specific content name meet the threshold, the content is labeled as MFR content, and it is recommended to be cached at the node associated with maximum number of neighbor nodes along the data routing path. According to EHCP, if a centrality node is selected as an edge node, the same content will be discarded to be cached at the same node again, and only content can be cached once at a node. Algorithm 4 represents the selection of the centrality node.

EHCP is diagrammatically presented in Figure 1 in which User U1 sends out a request to download content C1. As the request for content C1 is reached at node N9, the node N9 instantly responds to the request by sending the required content to the user U1, and a copy of content C1 is cached at edge node N3 to fulfill the requirements of subsequent requests. The content C1 is cached only at edge node N3 because it was downloaded once and it is not suggested as the MRF content. On the other hand, three requests from user U1, user U2, and user U3 are generated to download the content C2. As these requests for content C2 are received at the provider node N9, in response, the node N9 sends the content C2 toward the users U1, U2, and U3. However, the caching of content C2 is done at the centrality node. The reason is that the number of received requests for content C2 is equal to the threshold. Therefore, it is suggested as MFR content because the content C2 has received three requests that identify the content C2 is MFR as given in Algorithm 2. Therefore, the content C2 is selected to be cached at the node associated with a maximum number of neighbor nodes. Consequently, the first copy of C2 is cached at node N5. In addition, the second copy of C2 is cached at edge nodes N1, N3, and N4, as shown in Figure 1. After a while, another request is sent from user U4 to download content C2, the content C2 is found at node N5. Consequently, the node N5 becomes a provider and immediately sends the required content C2 to user U4, and a copy of content C2 is cached at edge node N8. Therefore, the entire subsequent requests from user U4 are accomplished from edge node N8, as shown in Figure 1.

The Least Replacement Policy (LRU) is adopted due to its flexible nature to delete the redundant cached contents to avoid the unnecessary usage of the cache. As the cache of a network node becomes full, the LRU is followed to remove the least recently used content to accommodate the new coming content. Through this way, the following goals are achieved: the cache hit ratio is improved, the content redundancy is kept at minimum, and the diversity and hop-decrement are improved.

V. PERFORMANCE EVALUATION AND METRICS

For the evaluation of the EHCP Caching strategy, we have conducted extensive simulation by using the SocialCCNSim simulator, in which each node is associated with cache storage. SocialCCNSim simulator was developed to evaluate the different categories of NDN-based caching strategies. For the current study, the caching performances of the strategies are evaluated in terms of cache-hit ratio, diversity ratio, hop decrement, and content redundancy. The EHCP strategy is compared with related NDN-based caching strategies such as ProbCache, LPD, and CCAC. We have selected 0.7, 1.0, 1.5, and 2.0 as Zipf Alpha parameters [34]. The Zipf distribution is required to choose a particular type of traffic as content popularity model. For the present study, four categories of Zipf alpha are suggested to be examining the performances of EHCP through Abilene, GEANT, Tiger, and DTelekom topologies. The social network generator (SONETOR) is used to generate the traffic which is used in Social network graph as Facebook [35].

In social networks, a large number of users are associated with a social network graph in which 4,039 nodes are linked with 88,234 friends across the network that known as edges of the network [36]. For the precise simulations, cache sizes are set to 1GB to 10GB (100 to 1000 elements) to check the consequences of EHCP with different cache sizes. In addition, diverse topologies are selected to check the efficiency of EHCP using different parameters. The selected parameters for the simulation environment are shown in Table 3.

To test the effectiveness of content placement strategies we have used diverse metrics for the evaluation of EHCP. For the present study the four metrics (the content Diversity Ratio, Cache Hit Ratio, Hop-Decrement Ratio, and Content Redundancy) are selected to evaluate the efficiency of the proposed EHCP strategy.

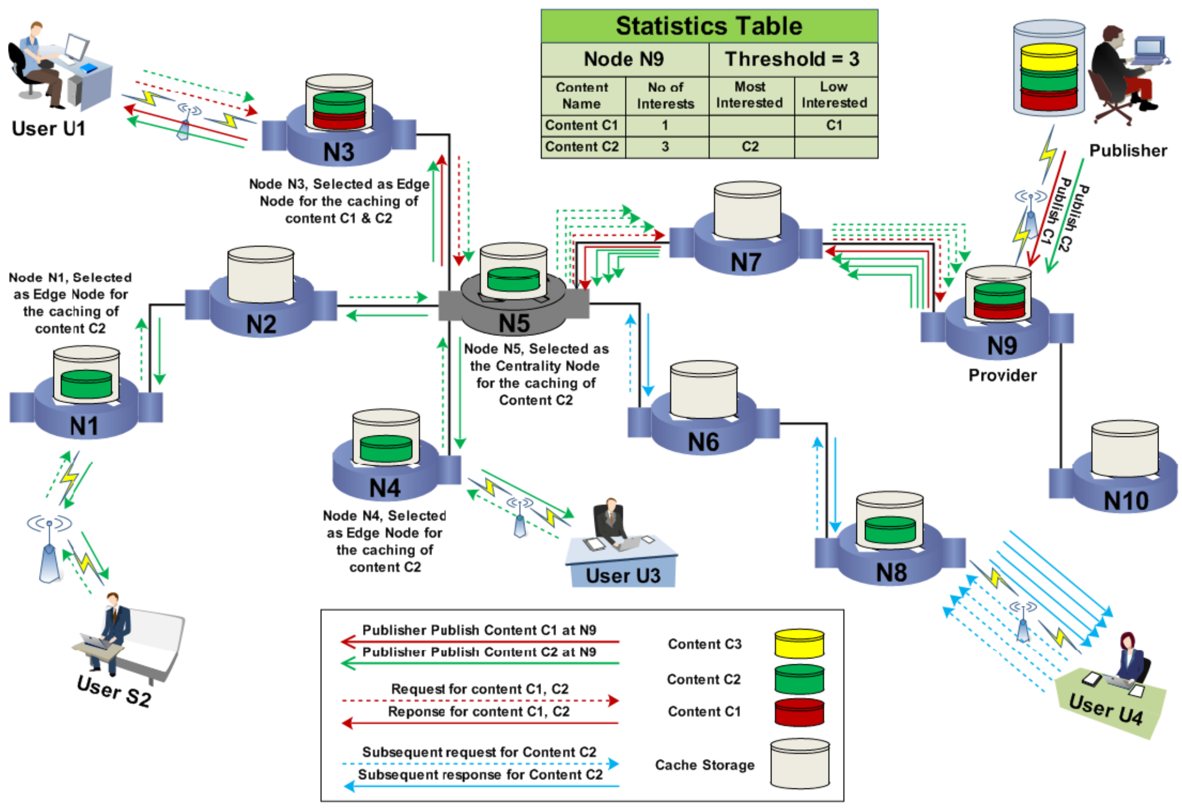
Statistics Table
Node N9
Threshold = 3
Content Name
No of Interests
Most Interested
Low Interested
Content C1
1
C1
Content C2
3
C2
User U1
N3
Node N3, Selected as Edge Node for the caching of content C1 & C2
Publisher
Publish C1
Publish C2
N7
Node N1, Selected as Edge Node for the caching of content C2
N2
N5
N9
Provider
N1
Node N5, Selected as the Centrality Node for the caching of Content C2
N6
N4
N10
Node N4, Selected as Edge Node for the caching of content C2
User U3
N8
User S2
User U4
Publisher Publish Content C1 at N9
Publisher Publish Content C2 at N9
Request for content C1, C2
Reponse for content C1, C2
Subsequent request for Content C2
Subsequent response for Content C2
Content C3
Content C2
Content C1
Cache Storage

FIGURE 1. Efficient hybrid content placement.

TABLE 3. Simulation description value.

Parameter	Value/Description
Simulation time	12 hours
Chunk Size	10MB
Cache Size	100-1000 elements or 1GB-10GB
Catalog Size	10^5
Alpha value	0.7, 1.0, 1.5, 2.0
Topology	Abilene, GEANT, Tiger, DTelekom
Replacement Policy	Least Recently Used (LRU)
Simulator	SocialCCNSim
Traffic Source	SONETOR (Facebook [39])
Performance Metrics	Diversity Ratio, Cache Hit Ratio Hop-Decrement and Redundancy

A. CONTENT DIVERSITY

The amounts of diverse contents reside in a network. In NDN caching, the different types of contents need to store within route's cache along the data routing path. Diversity can be calculated using the following equation:

$$D_c = \frac{\bigcup_{v=1}^{V} X_v}{\sum_{v=1}^{V} C_v} \tag{4}$$

where D_c shows the content diversity, V represents the total number of nodes, $\bigcup_{v=1}^{V} X_v$ represents the number of unique content X cached at v nodes, $\sum_{v=1}^{N} C_v$shows the total amount of cache storage at all nodes, and C_v shows the cache storage of a node v.

The intention of EHCP is to minimize a large number of homogeneous content' replications along the data delivery path to accommodate the newly requested miscellaneous contents. In fact, the replication of analogous contents increases the redundant caching operations which upsurge the content redundancy that causes network congestion and high traffic loads along the network channels. EHCP also increases the accommodation of heterogeneous contents through caching miscellaneous contents along the data routing path to increase the overall diversity ratio [38], [39]. In fact, diversity can increase by caching distinctive contents on discriminatory nodes in a network. Diversity provides the ratio between the number of unique and similar contents cached at the same location. From the results given in Figure 2, it is depicted that ProbCache shows minimum level of diversity ratio. The reason is that the replication of homogeneous contents is higher in ProbCache. According to ProbCache, all the transmitted contents are cached at maximum number of nodes along the data routing path that reduces the diversity ratio in ProbCache because of redundant caching operations. With increasing the cache size from 1GB to 10GB (100 to 1000 element), ProbCache still performs poor in terms to achieve high diversity ratio.

LPD performs better to some extent as it caches content at a downstream node as a first caching operation that increases the chance to accommodate more contents along the data delivery path. CCAC is failed to achieve a better Diversity ratio. The reason is that, CCAC cache all the selected contents

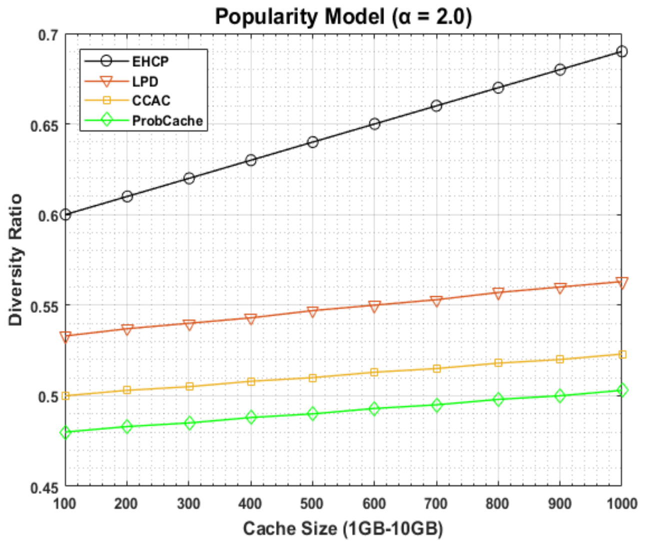

FIGURE 2. Diversity ratio on abilene topology.

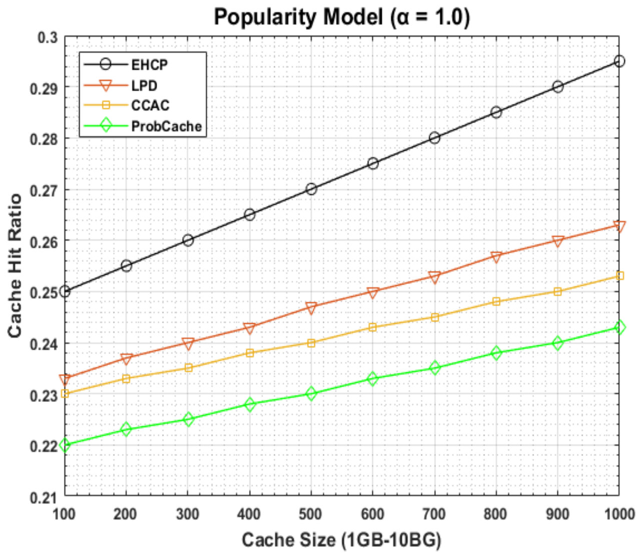

FIGURE 3. Cache hit ratio on GEANT topology.

at all the nodes along the data routing path. However, EHCP boosts up the diversity ratio because it does not allow similar contents to be cached at multiple locations alongside the data delivery path. As the cache size expands from 100 to 1000 elements, it seems that EHCP still performs better than benchmark strategies as illustrated in Figure 2.

B. CACHE HIT RATIO

Cache-Hit ratio represents the quantity in the average of existing contents hits (found) as the requests are sent [40]. Cache-Hit can be defined by the following equation

$$H_R = \frac{\sum_{n=1}^{n} Hit_n}{\sum_{n=1}^{n} (Hit + Miss)} \tag{5}$$

where H_R shows the hit ratio. Numerous networking and database researches have been prioritized the cache-hit ratio in terms of improving network performance. For a specific NDN-based caching strategy, the cache-hit ratio can be obtained by calculating the total number of hits and misses at a node. For the benefit of comparison, the consequence regarding the cache-hit ratio is pictorially represented in Figure 3. As the cache size expand (from 100 to 1000 elements), the performance of the network noticed to have better performance excluding the ProbCache, because of its probabilistic nature to replicate a content at all nodes. Figure 3 shows the obtained results regarding cache-hit ratio using GEANT topology with $\alpha = 1.0$ that is considered as file sharing content.

ProbCache represents a low level of cache-hit ratio because it caches similar content at all the nodes. However, EHCP shows better performance in terms of hit ratio, and it is competing with LPD and CCAC to some extent because in LPD and CCAC the amount of homogeneous contents is higher than EHCP. The reason is that the LPD and CCAC strategies create similar content replications along the data routing path. If the path is very small, it makes trouble to cache the new contents. Hence, several evictions are required to accommodate the new content, which decreases the cache-hit ratio because all the new requests will accomplish from the remote providers that increase the distance between consumer and cached content. The EHCP performed better as compared to other caching strategies with different cache size as shown in Figure 3. In fact, EHCP caches heterogeneous contents close to the users and consequently the subsequent requests are satisfied from the nearest nodes. Therefore, the EHCP has performed better than benchmark strategies with restricted cache sizes in hand.

C. REDUNDANCY

Content redundancy is a significant metric to assess homogeneous caching operations done by similar content along the data delivery path. The redundancy can be determined by the following equation:

$$R_c = \sum_{i=1}^{n} Rc_i, \tag{6}$$

where R_c represents the content redundancy, Rc_i shows the redundant content at a specific location. ProbCache produces huge amount of duplications that affects the whole performance and increases the congestion. Figure 4 illustrates the noticeable cause of numerous caching operations is made by ProbCache due to its high level of redundancy. CCAC shows reduced redundancy than ProbCache because in CCAC, the caching decision is made by the popular contents and the content is not supposed to be cached if the popularity of that content is decreased. Consequently, CCAC increases the chance to accommodate heterogeneous content to some extent. LPD shows a low level of redundant caching operations due to caching the content at restricted positions (downstream node and edge node). The reason is that, if the content is cached at the edge node, the cache storage along the data routing path remains unallocated that increases the free storage to accommodate new contents. As compare to

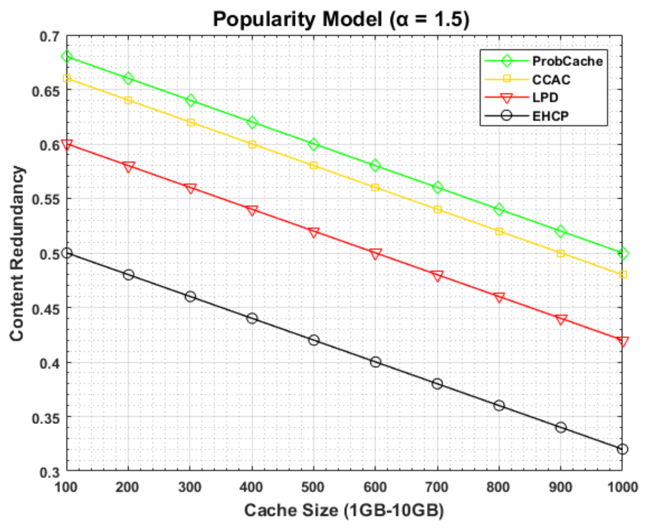

FIGURE 4. Content redundancy on tiger topology.

ProbCache, CCAC, LPD, EHCP performs better in terms of redundant caching operations because of its heterogeneous nature to execute limited caching operations alongside the data routing path. It considers all the contents to be cached, but it prioritizes the contents according to their frequency count.

In fact, it accommodates the most frequently requested content as well as less frequently requested contents that increase the number of diverse contents cache alongside the data routing path. Figure 4 presents the performance in terms of redundancy after tested in a simulation environment using Tiger topology with 1GB to 10GB cache size. The outcome shows the EHCP performs better in terms of redundancy than state-of-the-art comparing strategies.

D. HOP-DECREMENT

Hop-decrement can be defined as the number of hops (nodes) which are needed to be traversed by a use's request between the cached content (where the hit occurs) and the user. When a user requests for some content, the request needs to traverse through a number of hops to reach the content to download it. Sometimes the request finds a copy of the required content from any of the network nodes that appears on the path. Otherwise, the request needs to go to the publisher to find the appropriate content. The EHCP strategy reduces the maximum number of hops between the required content and the users by caching content close to the users as at centrality nodes and edge nodes. The hop-decrement [41] can be calculated using the stretch, which is one of the significant metrics to measure the NDN caching performance as given by the following equation;

$$S = \frac{\sum_{i=1}^{n} \mathcal{H}_c}{\sum_{i=1}^{n} \mathcal{H}_t} \tag{7}$$

$$\mathcal{H}_d = 1 - S \tag{8}$$

where S shows the stretch, $\mathcal{H}_c$ represents the number of hops traveled by content c between the user and provider (where the cache hit occur), $\mathcal{H}_t$ shows the total number of hops, and $\mathcal{H}_d$ represents the hop-decrement.

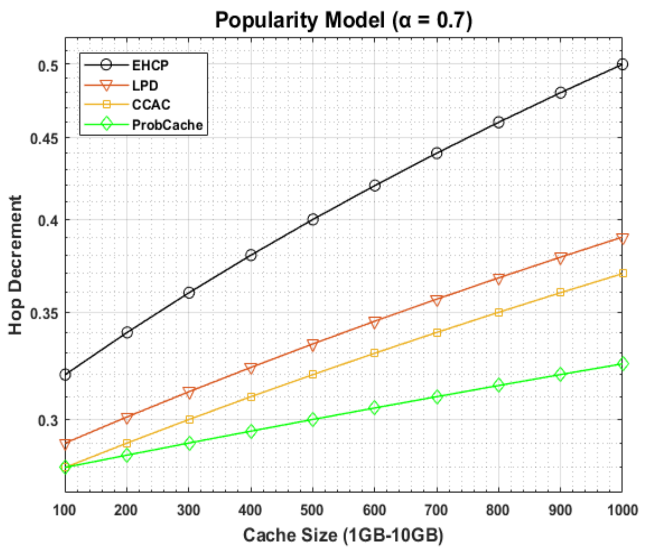

FIGURE 5. Hop-decrement on DTelekom topology.

In Figure 5, the results are obtained using DTelekom topology. From the given result, it can be concluded that the ProbCache and CCAC show poor performance because of both strategies cache all the contents at all on-path nodes. As we know, the cache capacity is much smaller as compared to the number of contents to be cached. Therefore, a large number of eviction operations are accomplished to accommodate the new coming content, and when a request for new content is received to download some content, it needs to forward to the remote location to get the appropriate content. Consequently, hop decrement is increased.

However, LPD is performed better to some extent because it decreases the hop decrement only for most frequently requested contents and when the requests for least frequently content are generated the contents are not found at intermediate nodes. Subsequently, the requests need to traverse the several hops to find suitable contents from a remote location and hence, the hop decrement is increased. On the other hand, the EHCP performs better than all other strategies because it manages the both most frequently and least frequently requested content that increases the hop decrement. Consequently, the EHCP is performed better to achieve a high level of hop decrement and it performed better with different cache sizes (1GB to 10GB) than comparing strategies, as shown in Figure 5.

VI. RESULTS AND DISCUSSION

The NDN caching is basically divided into two sections. The first section is related to the placement of the user requested content while the second section is related to the replacement of content as the cache overflows to accommodate the newly arriving contents. In this study, we have proposed a content placement strategy to enhance the content caching mechanism in NDN. Moreover, the present strategy focuses on the

TABLE 4. Numerical results of the simulation.

Performance Metrics	Strategies	1GB	2GB	3GB	4GB	5GB	6GB	7GB	8GB	9GB	10GB
Diversity Ratio ($\alpha = 1.0$) Abilene Topology	ProbCache	0.48	0.483	0.485	0.488	0.49	0.493	0.495	0.498	0.50	0.503
	CCAC	0.50	0.503	0.505	0.508	0.51	0.513	0.515	0.518	0.52	0.523
	LPD	0.535	0.538	0.54	0.543	0.545	0.548	0.55	0.555	0.56	0.565
	EHCP	0.60	0.61	0.62	0.63	0.64	0.65	0.66	0.67	0.68	0.69
Cache Hit Ratio ($\alpha = 2.0$) GEANT Topology	ProbCache	0.22	0.222	0.224	0.226	0.23	0.232	0.234	0.236	0.238	0.24
	CCAC	0.23	0.232	0.234	0.236	0.24	0.243	0.245	0.248	0.25	0.253
	LPD	0.233	0.235	0.238	0.24	0.243	0.245	0.248	0.25	0.255	0.262
	EHCP	0.25	0.255	0.26	0.265	0.27	0.275	0.28	0.285	0.29	0.295
Content Redundancy ($\alpha = 1.5$) Tiger Topology	ProbCache	0.68	0.66	0.64	0.62	0.60	0.58	0.56	0.54	0.52	0.50
	CCAC	0.66	0.64	0.62	0.60	0.58	0.56	0.54	0.52	0.50	0.48
	LPD	0.60	0.58	0.56	0.54	0.52	0.50	0.48	0.46	0.44	0.42
	EHCP	0.50	0.48	0.46	0.44	0.42	0.40	0.38	0.36	0.34	0.32
Hop Decrement ($\alpha = 0.7$) DTelekom Topology	ProbCache	0.28	0.285	0.29	0.295	0.30	0.305	0.31	0.315	0.32	0.325
	CCAC	0.28	0.29	0.30	0.31	0.32	0.33	0.34	0.35	0.36	0.37
	LPD	0.29	0.30	0.31	0.32	0.33	0.34	0.35	0.36	0.375	39
	EHCP	0.33	0.35	0.37	0.39	0.41	0.43	0.45	0.47	0.49	0.51

most significant metrics, such as content diversity ratio, cache hit ratio, content redundancy, and hop decrement. These are the most important metrics used by a content placement strategy to check the effectiveness of the proposed model. These metrics usually use to evaluate the performance of the proposed strategy by comparing it with the existing strategies. For this reason, we have selected three strategies named as ProbCache, CACC, and LPD, to evaluate our proposed model. According to the achieved consequences, the proposed EHCP is performed better than benchmark strategies in terms to achieve the following goal;

- Higher cache hit Ratio
- Enhanced hop decrement Ratio
- Enhanced content diversity Ratio
- Reduced content redundancy Ratio

From Figure 2, Figure 3, Figure 4, and Figure 5 we can say the EHCP performed slightly better with small cache size. However, EHCP achieved better results with large cache size. LPD and CCAC strategies are increased the homogeneous content replications through caching the similar content at multiple nodes, which cause a high level of redundant contents' replications and reduces the content diversity. In these strategies, the similar contents are cached at all the nodes along the data delivery path that minimizes the cache hit ratio and increases the distance between the user and publisher.

The reason is that all the transmitted contents are cached at all nodes and when the cache of those nodes is full, it requires several evictions to accommodate new contents. Consequently, when a user request is received for old but popular content, it needs to traverse several hops to download required content from the remote publisher that increases the path length and consequently the hop-decrement is reduced, and the minimum cache-hit ratio is accomplished. Probabilistic cache and CCAC caching strategies aim to distribute cache capacity to improve the overall cache performance, but these strategies increase the inefficient usage of the resource (cache) by caching the similar contents at multiple nodes along the unique data routing path. Moreover, if the publisher-subscriber path is small, the redundancy is maximized due to large replications of similar contents.

ProbCache cannot accomplish its goals because it seeks to reduce memory consumption by mitigating the content redundancy, but it does not overcome the mentioned issues along the short data delivery path. It does not provide the positioned distinction while the cache of a node is overflowed and the popular content needed to be cached at the critical location that produces the low ratio of cache-hit because there is a possibility of caching contents far from the user that increases the distance between the publisher and the user [31]. ProbCache and CCAC objective to tackle the problems related to the inefficient usage of the resources in a different way. However, both strategies try to share the resources (cache) fairly among the contents, but it seems unable to achieve their goals completely with a small publisher-subscriber path [38]. These strategies introduce several parameters that have no content distinction and need to compute these parameters arbitrarily for each user request and content at all the nodes. Additionally, these strategies need to update requests and contents in the form of Time Since Inspection (TSI) Time Since Birth (TSB) that increase the computational cost and reduces the cache-hit [31].

In Table 3, the numerical results are presented with different cache sizes from 1GB to 10GB on cache-hit ratio, content diversity, hop decrement, and redundancy. It is clear from these results, the EHCP strategy performed better as compared to LPD, CCAC, and ProbCache with respect to all selected parameters with all caches size (1GB to 10GB) in hand.

VII. CONCLUSION

In this paper, we have developed an NDN-based content placement strategy named as Efficient Hybrid Content

Placement (EHCP). It is developed to diminish the multiple replications of similar contents alongside the data routing path. Due to the limited cache capacity, the existing strategies could not perform better to improve the overall caching performance. However, the proposed strategy increases the content diversity ratio through caches the heterogeneous content in-network and has achieved enhanced caching performance. The goal of EHCP is to increase content diversity by reducing the redundant content replications, and it improves the overall cache-hit ratio. We have assessed our strategy in a simulation platform and evaluated the effectiveness of proposed EHCP against some state-of-art caching strategies in a simulation environment. The consequences showed that proposed strategy has achieved better results while the cache size is restricted.

REFERENCES

[1] A. Banerjee, X. Chen, J. Erman, V. Gopalakrishnan, S. Lee, and J. Van Der Merwe, "MOCA: A lightweight mobile cloud offloading architecture," in *Proc. 8th ACM Int. Workshop Mobility Evolving Internet Archit.*, 2013, pp. 11–16.

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

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

[4] H. Jin, D. Xu, C. Zhao, and D. Liang, "Information-centric mobile caching network frameworks and caching optimization: A survey," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, p. 33, 2017.

[5] X. Yang, M. I. Levi, E. A. Lezaun, A. A. G. S. Mendoza, D. G. Guillen, A. P. Cao, M. Burrel, and D. C. Olesti, "Distributed health-check method for Web caching in a telecommunication network," U.S. Patent 9 948 741 B2, Apr. 17, 2018.

[6] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache 'less for more' in information-centric networks (extended version)," *Comput. Commun.*, vol. 36, pp. 758–770, Apr. 2013.

[7] M. A. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "Caching in named data networking for the wireless Internet of Things," in *Proc. Int. Conf. Recent Adv. Internet Things (RIoT)*, Apr. 2015, pp. 1–6.

[8] M. M. F. Hamdi, A. Habbal, N. H. Zakaria, and S. Hassan, "Evaluation of caching strategies in content-centric networking (CCN) for mobile and social networking environment," *J. Telecommun., Electron. Comput. Eng.*, vol. 10, pp. 1–6, Jul. 2018.

[9] C. Fang, F. R. Yu, T. Huang, J. Liu, and Y. Liu, "A survey of green information-centric networking: Research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1455–1472, 3rd Quart., 2015.

[10] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Comput. Netw.*, vol. 51, no. 4, pp. 921–960, Mar. 2007.

[11] H. Wu, J. Li, T. Pan, and B. Liu, "A novel caching scheme for the backbone of named data networking," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2013, pp. 3634–3638.

[12] X.-D. Cui, T. Huang, L. Jiang, L. Li, J.-Y. Chen, and Y.-J. Liu, "Design of in-network caching scheme in CCN based on grey relational analysis," *J. China Universities Posts Telecommun.*, vol. 21, pp. 1–8, Apr. 2014.

[13] C. Bernardini, T. Silverston, and O. Festor, "A comparison of caching strategies for content centric networking," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2015, pp. 1–6.

[14] C. Fang, H. Yao, Z. Wang, W. Wu, X. Jin, and F. R. Yu, "A survey of mobile information-centric networking: Research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2353–2371, 3rd Quart., 2018.

[15] C. Bernardini, T. Silverston, and O. Festor, "MPC: Popularity-based caching strategy for content centric networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2013, pp. 3619–3623.

[16] Y. Li, T. Zhang, X. Xu, Z. Zeng, and Y. Liu, "Content popularity and node level matched based probability caching for content centric networks," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Jul. 2016, pp. 1–6.

[17] G. S. Paschos, G. Iosifidis, M. Tao, D. Towsley, and G. Caire, "The role of caching in future communication systems and networks," May 2018, *arXiv:1805.11721*. [Online]. Available: https://arxiv.org/abs/1805.11721

[18] Q. Chen, R. Xie, F. R. Yu, J. Liu, T. Huang, and Y. Liu, "Transport control strategies in named data networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2052–2083, 3rd Quart., 2016.

[19] Q. N. Nguyen, M. Arifuzzaman, T. Miyamoto, and S. Takuro, "An optimal information centric networking model for the future green network," in *Proc. IEEE 12th Int. Symp. Auton. Decentralized Syst.*, Mar. 2015, pp. 272–277.

[20] Q. N. Nguyen, M. Arifuzzaman, K. Yu, and T. Sato, "A context-aware green information-centric networking model for future wireless communications," *IEEE Access*, vol. 6, pp. 22804–22816, 2018.

[21] 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.*, 2012, pp. 55–60.

[22] M. A. Naeem and S. A. Nor, "A survey of content placement strategies for content-centric networking," in *Proc. AIP Conf. Proc.*, 2016, Art. no. 020078.

[23] H. Yao, C. Fang, Y. Guo, and C. Zhao, "An optimal routing algorithm in service customized 5G networks," *Mobile Inf. Syst.*, vol. 2016, Dec. 2016, Art. no. 6146435.

[24] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2847–2886, 4th Quart., 2016.

[25] I. Psaras, W. K. Chai, and G. Pavlou, "In-network cache management and resource allocation for information-centric networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 11, pp. 2920–2931, Nov. 2014.

[26] Y. Qin, W. Yang, and W. Liu, "A probability-based caching strategy with consistent hash in named data networking," in *Proc. 1st IEEE Int. Conf. Hot Inf.-Centric Netw. (HotICN)*, Aug. 2018, pp. 67–72.

[27] G. Panwar, R. Tourani, T. Mick, A. Mtibaa, and S. Misra, "DICE: Dynamic multi-RAT selection in the ICN-enabled wireless edge," in *Proc. Workshop Mobility Evolving Internet Archit.*, 2017, pp. 31–36.

[28] S. Rahel, A. Jamali, and S. El Kafhali, "Energy-efficient on caching in named data networking: A survey," in *Proc. 3rd Int. Conf. Cloud Comput. Technol. Appl. (CloudTech)*, 2017, pp. 1–8.

[29] F. Qazi, O. Khalid, R. N. B. Rais, I. A. Khan, and A. ur Rehman Khan, "Optimal content caching in content-centric networks," *Wireless Commun. Mobile Comput.*, vol. 2019, Jan. 2019, Art. no. 6373960.

[30] *Cisco*. [Online]. Available: https://qwilt.com/events/cisco-live-2016/

[31] 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, May 2014.

[32] C. Bernardini, *Stratégies de Cache Basées Sur la Popularité Pour Content Centric Networking*. Université de Lorraine, 2015.

[33] S. Saha, A. Lukyanenko, and A. Ylä-Jääski, "Cooperative caching through routing control in information-centric networks," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 100–104.

[34] J. Ren, W. Qi, C. Westphal, J. Wang, K. Lu, S. Liu, and S. Wang, "Magic: A distributed max-gain in-network caching strategy in information-centric networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr./May 2014, pp. 470–475.

[35] P. K. Sharma, S. Rathore, and J. H. Park, "Multilevel learning based modeling for link prediction and users' consumption preference in online social networks," *Future Gener. Comput. Syst.*, vol. 93, pp. 952–961, Apr. 2017.

[36] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 539–547.

[37] I. Cantador, P. L. Brusilovsky, and T. Kuflik, *Second Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011)*. New York, NY, USA: ACM, 2011.

[38] M. Dräxler and H. Karl, "Efficiency of on-path and off-path caching strategies in information centric networks," in *Proc. IEEE Int. Conf. Green Comput. Commun. (GreenCom)*, Nov. 2012, pp. 581–587.

[39] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," *Comput. Netw.*, vol. 57, no. 16, pp. 3128–3141, 2013.

[40] T. Mick, R. Tourani, and S. Misra, "MuNCC: Multi-hop neighborhood collaborative caching in information centric networks," in *Proc. 3rd ACM Conf. Inf.-Centric Netw.*, 2016, pp. 93–101.

[41] D. Rossi and G. Rossini, "On sizing CCN content stores by exploiting topological information," in *Proc. IEEE INFOCOM Workshops*, Mar. 2012, pp. 280–285.

• • •