IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 26, 2017, accepted February 20, 2017, date of publication April 13, 2017, date of current version June 7, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2694045

Caching Strategy Based on Hierarchical Cluster for Named Data Networking

HUAN YAN[1], DEYUN GAO[1], WEI SU[1], CHUAN HENG FOH[2], (Senior Member, IEEE), HONGKE ZHANG[1], AND ATHANASIOS V. VASILAKOS[3]

[1]School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China
[2]5G-IC, Department of Electrical and Electronic Engineering, Institute for Communication Systems, University of Surrey, Surrey GU1 2UX, U.K.
[3]Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, SE-931 87 Skelleftea, Sweden

Corresponding author: Deyun Gao (gaody@bjtu.edu.cn)

This work was supported in part by the 973 Program under Grant 2013CB329100 and in part by NSFC under Grant 61232017 and Grant 61272504.

ABSTRACT The in-network caching strategy in named data networking can not only reduce the unnecessary fetching of content from the original content server deep in the core network and improve the user response time, but also ease the traffic in the core network. However, challenges exist in in-network caching, such as the distributed locations of storage and relatively small cache space which limit the hit rate, and the cache management introduces further overhead. In this paper, we propose a two-layer hierarchical cluster-based caching solution to improve in-network caching efficiency. A network is grouped into several clusters, then, a clusterhead is nominated for each cluster to make caching decision. The clustering approach offers scalability and permits multiple aspects of inputs to be used for decision making. Our solution jointly considers the location and content popularity for caching. We implement our strategy in ndnSIM and test it on GEANT-based network and AS3967 network. Our simulation results show significant improvement over its peers.

INDEX TERMS Hierarchical cluster, betweenness centrality, caching strategy, named data networking.

I. INTRODUCTION

In Named Data Networking (NDN), caching the requested content in the routers along the delivery path is a unique feature [1]–[3]. In-network caching allow users to obtain the contents from nearer intermediate routers, reducing the need for content fetching from the servers often located deep in the network [4]. The benefits of in-network caching are obvious. Firstly, intermediate routers can share the responsibility of providing contents which lighten the load of the original content servers. Secondly, peer-to-peer traffic involves shorter paths and reduces the chance of congestion. Finally, the response time and the transmission overhead for fetching contents are reduced because the required contents can be fetched from the nearest cache instead.

In order to take full advantage of above benefits, an appropriate caching strategy is needed. A simple caching strategy of NDN named Leave Copy Everywhere (LCE) caches every content in the intermediate routers. While it is easy to implement, caching all contents is impractical given the limited cache space in routers. Besides, blindly caching every content increases the chance of caching contents that will not be requested again, leading to inefficient occupation of the limited cache space. Thus, a careful design of caching strategy is needed to ensure high hit rate given limited cache space.

To manage distributed locations of cache storage, we use a hierarchical clustering approach to manage the in-network caching. Clustering approach is common in network design to manage distributed entities in a network, such as those in Mobile Ad hoc NETworks (MANETs) [5], [6], Wireless Sensor Networks (WSN) [7], [8] and Vehicular Ad hoc NETworks (VANETs) [9]. As clustering approach permits scaling of a potentially large network into several smaller autonomous groups as well as scoping of operational functions and message exchanges within a cluster, this approach generally offers high scalability and efficiency.

To take the advantage of clustering approach for management of distributed caching, we present a Hierarchical Clusterbased Caching (HCC) solution. Our design has a two-layer hierarchical clustering architecture. The routers in Core Layer are not used for caching so that they can focus on content routing. In Edge Layer, routers are designed to cache contents for prompt user responses. Furthermore, we introduce importance rating where nodes of higher (resp. lower)

importance rating in Edge Cluster cache more (resp. less) popular contents. Clusterhead has a responsibility to collect and allocate the information of node importance based on betweenness centrality, content popularity and probability matrix in its cluster. In additional, all the important nodes execute their respective caching decision whether a content should be cached, and if so, where. Our proposed HCC solution is implemented in ndnSIM [10] considering two different network topologies. We show that HCC outforms other strategies in several aspects.

The part of initial work has been published in our previous conference paper [11]. In this paper, we made significant extensions in three aspects. Firstly, we present the computation process of probability matrix and its elements, content caching/update strategy, and cluster-based routing protocol. Secondly, both GEANT-based network and AS3967 network are adopted as simulation topologies. And we demonstrate that the trends of simulation results are similar. Thirdly, we analyze the performance of five caching strategies affected by content storage and content popularity distribution that are two important factors of caching.

In the next section we briefly introduce NDN and some related research works. Section III presents the architecture of our proposed HCC, clusterhead election and establishment of hierarchical cluster. In Section IV, the hierarchical cluster-based caching strategy is described. Section V presents the performance of our proposed method and highlights the performance advantages compared with its peers. Finally in Section VI, important conclusions are drawn.

II. RELATED WORKS

As shared by Cha *et al.* [12], 80% of the requests can be satisfied if a cache stores only 10% of long-term popular videos in YouTube. The study has motivated many research works focusing on proposing new caching strategies to improve the effectiveness of caching. Generally, a caching strategy solves two problems, namely which locations are most appropriate to cache contents and which contents should be cached. In the following, we survey existing related works that focus on decision making on location selection and/or content popularity individually within a device or collaboratively within a network.

The simplest form of location selection strategy for caching is probability-based caching. A caching strategy named Prob(p) suggests to cache each content at every router along the delivery path with a certain probability [13]. It is stateless and it disregards of the location of routers in the network when making caching decision. To allow more cached contents nearer to the destination, Probabilistic Cache (ProbCache) proposes using higher probability to cache contents when a router is closer to the destination [14]. Similarly, Hop-based Probabilistic Caching (HPC) uses the number of the hops between the responder (content provider or intermediate router) and the router to determine the caching probability [15]. Leave Copy Down (LCD) [13] prefers to cache the copies in the next router which is along the downstream path of cache hit router. WAVE [16] is another approach similar to LCD. WAVE extends LCD by dividing contents into several chunks and caching them in the next downstream router with consideration of content popularity. A betweenness centrality-based caching algorithm (Betw) caches contents at the most important router through where most content delivery paths pass [17]. All the above-mentioned solutions mainly deal with the location for caching and pay little attention to the contents.

User request frequency is a common indicator for content popularity, and hence on the basis of location selection, a caching strategy with content popularity will help to decide which contents should be cached. Li *et al.* [18] present popularity-based caching algorithms using a hierarchical tree topology. The content popularity information is aggregated from end nodes to the root, and the caching decisions are spread from the top level to the bottom level. However, too much engagement from downstream routers for caching has led to high advertisement overhead. Without involvement in many routings, CRCache [19] introduces content caching based on correlation of content popularity and network topology. Its main drawback is the use of selected routers in network backbone rather than network edges where prompt response can be offered. MAx-Gain In-network Caching (MAGIC) [20] introduces a utility called cache gain based on content popularity, hop reduction and cache replacement penalty to consider content caching. Contents are judged for caching using the utility to maximal the cache gain. However, the computational complexity of MAGIC is very costly because it needs to calculate all three indexes of each content at each node.

Instead of individual decision making for caching, collaborative caching strategy attempts to introduce collaboration among routers to make caching decision for lower redundancy in content caching. This strategy generally involves in a certain message exchanges among devices to jointly make caching decision. Age-based cooperative caching [21] spreads popular contents to the network edge by dynamically changing the caching time of contents in an implicitly cooperation manner. In [22], an intra-AS cache cooperation scheme is proposed to allow neighbor nodes to eliminate redundancy after caching content and collaborate in serving each other's requests. Intra-Domain cooperative caching (IDCC) scheme in [23] combines probabilistic caching and hierarchical caching. In IDCC, cached content advertisements are diffused in the intra-domain to minimize cache redundancy for improved cache utilization. Dai *et al.* [24] investigate the capacity provisioning problem and propose a collaborative hierarchical caching mechanism accordingly. The main drawback of collaborative caching approach is the need for message exchanges which introduces additional overheads and scalability issue.

Our proposed HCC solution jointly considers location selection and content popularity for caching. HCC further takes collaborative caching approach allowing devices to exchange information for more effective caching and less

redundancy. However, as a collaborative caching approach, the main challenge in the design is the management of overheads while maintaining high caching effectiveness. We address this by introducing a hierarchical cluster-based architecture to reduce communication overhead for scalability. Our proposed collaborative probability caching further reduces caching redundancy for effective use of cache memory storage.

III. HIERARCHICAL CLUSTER-BASED ARCHITECTURE

In this section, we describe the design of hierarchical cluster-based architecture. We first introduce the hierarchical cluster-based structure, followed by a variable weighted-based clustering algorithm for clusterhead selection. Finally, we present the procedures of cluster establishment.

A. HIERARCHICAL CLUSTER-BASED STRUCTURE

Hierarchical topology has a broad range of application in many existing systems (e.g., multicast routing, content broadcasting). Based on [25], we consider a typical network topology with a two layers hierarchical structure which are called Core Layer and Edge Layer. Core Layer has large bandwidth and provides fast packet transportation among Autonomous System (AS) areas, which is constituted by the core routers in wide area network (WAN). Edge Layer provides user access to the network via edge routers.

Our design of caching role has the following principles. In Core layer, all routers do not perform caching. This consideration is to avoid disruption of routing function in the Core Layer. In Edge Layer, routers are allowed to cache contents depending on their importance. Generally, more important routers in Edge Layer should cache more popular contents, while less important routers cache less popular contents, and non-important routers should not perform any caching.

Routers exchange information to collaboratively and dynamically adjust their caching roles. The management of caching and message exchange is done by clustering. For scalability, Edge Layer may contain one or more clusters depending on the size of the Edge Layer. These clusters are called Edge Clusters. Fig. 1 shows an example of the hierarchical cluster-based network topology for caching.

Each Edge Cluster contains three types of nodes, namely clusterhead, gateway and member. Routers in Edge Cluster provide connectivity to content providers and content requesters. Content providers and requesters are not part of the Edge Cluster, and thus they do not participate in cluster formation nor the management. This design allows content providers and requesters to move freely from one location to another without affecting the involved clusters. The following summarizes the main role of clusterhead, gateway and member.

- Clusterhead is a cluster controller which is elected by other cluster members. Section III-B explains the procedure of clusterhead election. Its main responsibility is to gather and maintain the betweenness centrality of all the nodes and popularity of contents requested in the cluster. Furthermore, it calculates probability matrix and distributes the outcome within its cluster.
- Gateway is a border router between two clusters. It can participate in the clusterhead election and forward Interest and Data packets among different clusters.
- Member is the remaining router neither a clusterhead nor a gateway. All members have a right to vote for the clusterhead. Members will cache temporary contents according to the probability and content popularity. More detail about the operation is given in Section IV. Besides, some members will also act as access routers communicating with content providers and content requesters.

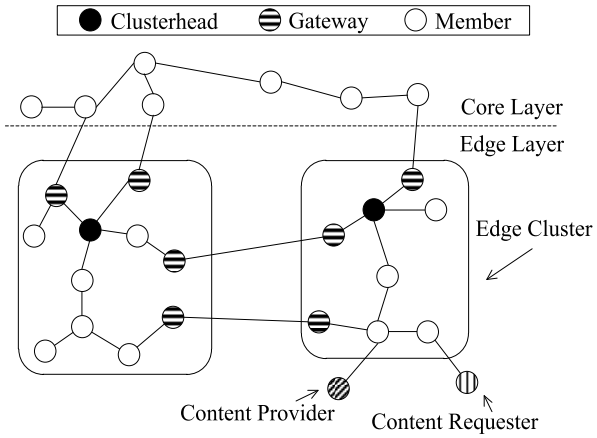


FIGURE 1. The hierarchical cluster-based network topology.

Clusters are named by their corresponding layers, such as, a cluster with an ID of i located at the Edge Cluster is represented by EC_i. A clusterhead of EC_i is named EC_i_H. The j-th gateway in EC_i is named $EC_i\_G_j$. The k-th cluster members in EC_i is named $EC_i\_R_k$. For Core Layer, there is only one cluster in the system without any clusterhead.

B. CLUSTERHEAD ELECTION

Our clustering approach is mainly based on Weighted-based Clustering Algorithm (WCA) [26]. In WCA, a weight function is defined to consolidate a number of factors together and produce a single value that rates each node. The node possesses the lowest value among all is chosen as the clusterhead. Designing for mobile ad hoc networks, WCA uses the following four factors to measure a node, which are degree-difference between the node degree of the measured node and the pre-defined node limitation, the sum of distances from the measured node to all neighbors, the average moving speed of the measured node, and finally the serving time of the measured node which is related to the remaining battery level.

We follow WCA for our clusterhead election. Due to different considerations of network scenarios and use cases, we use different factors for our computation. Considering a network of M nodes, we first label each node uniquely by an integer starting from 1. Labels of all nodes in the network

is collected in a set $\mathbf{V} = \{1, 2, ..., M\}$ and the cardinal of $\mathbf{V}$ is $|\mathbf{V}| = M$.

We consider the following three factors to measure each node, which are (i) the reciprocal of neighbor node degree $\frac{1}{d_v}$ of the measured node v, (ii) the average transmission time T_v to all reachable nodes from node v, and (iii) the weighted-based hops H_v of node v. The final score of a node is a weighted sum of all three factors defined in (1).

$$W_v = \alpha_1 \frac{1/d_v}{\overline{R_d}} + \alpha_2 \frac{T_v}{\overline{T}} + \alpha_3 \frac{H_v}{\overline{H}} \tag{1}$$

where α_1, α_2, and α_3 are the weighting factors, and $\alpha_1 + \alpha_2 + \alpha_3 = 1$. Similar to WCA, a node with the lowest overall value among all is chosen as the clusterhead for the cluster.

The first factor stated in (1) is the reciprocal of neighbor node degree $\frac{1}{d_v}$. The quantity d_v is the number of neighbors of node v. A smaller node degree reciprocal means larger number of neighbors. The system-wide average reciprocal of neighbor node degree, $\overline{R_d}$ can be calculated by $\frac{1}{M}\sum_{i=1}^{M}\frac{1}{d_i}$.

The second factor stated in (1) is the average transmission time to reachable nodes from node v. Precisely, T_v, is defined as $T_v = \frac{1}{|\mathbf{C_v}|}\sum_{j\in\mathbf{C_v}} t_{v,j}$, where $\mathbf{C_v}$ is the set of nodes connected to node v and $t_{v,j}$ is the transmission time from node v to any connected node j. We see that a smaller transmission time offers faster dissemination of information. Similarly, the system-wide average transmission time can be determined by $\overline{T} = \frac{1}{M}\sum_{i=1}^{M} T_i$.

The third factor stated in (1) is the weighted-based number of hops of node v, or H_v. This factor indicates the topology relationship in a cluster. In order to reduce the overhead of collecting and distributing caching information, the number of hops from clusterhead to any cluster nodes should be as low as possible. The quantity H_v can be determined by

$$H_v = \sum_{k=1}^{\infty} \frac{\beta_k \cdot k}{n_k} \tag{2}$$

where n_k is the number of k-hop neighbors of node v. The coefficient β_k describes the weight with higher weight for a lower hop, that is $\beta_1 > \beta_2 > \beta_3 > \cdots$. Here a heuristic parameter setting is $\beta_1 = \frac{1}{2}, \beta_2 = \frac{1}{3}, \beta_3 = \frac{1}{6}$ and $\beta_k = 0, k > 3$. Note that we truncate the summation at $k = 3$ to reduce the influence from faraway hops. The system-wide average weighted-based hops of the network is $\overline{H} = \frac{1}{M}\sum_{i=1}^{M} H_i$.

Based on the above equations, a node with the smallest overall weight value will be elected as a clusterhead. The selection favors a node with more neighbor nodes, a lower transmission time to other nodes, and a smaller number of hops to other nodes.

C. FORMATION OF HIERARCHICAL CLUSTER

The following describes the formation of cluster after the election of a clusterhead. To avoid management of a large cluster, we set a node limitation σ for each clusterhead which limits the number of nodes that a clusterhead can handle.

- Clusterhead sends a JOIN message to other nodes.
- These nodes reply PRE_JOIN message to clusterhead as a response to join the cluster.
- Clusterhead only selects the top σ nodes as the cluster members based on the rank of transmission time from clusterhead to other nodes, which is measured during the process of advertising JOIN and PRE_JOIN message. Then it sends CONF_JOIN message to the top σ nodes.
- The nodes reply JOIN_ACK message when they receive the first CONF_JOIN messages from clusterhead, and becomes a cluster member. Furthermore, some cluster members will become gateways if they have physical links connecting to other clusters. For subsequent CONF_JOIN messages from other clusterheads, it will send REF_JOIN message for refusing to join.
- The nodes which send JOIN_ACK messages will be added into the member list of clusterhead, and the nodes which send REF_JOIN messages should be deleted. If the number of member is less than σ, clusterhead continue to send CONF_JOIN message to other potential nodes.
- After the above process, if a clusterhead doesn't have any member, it will change its roles to a member. It may be aggregated into some adjacent clusters when the number of members in those clusters drops below σ. And a node which fails to join any cluster will do the same thing.

IV. HIERARCHICAL CLUSTER-BASED CACHING

In order to reduce caching redundancy and improve hit rate, betweenness centrality is introduced to calculate the importance of nodes for caching [17], [27]. The measure rates each node based on the frequency of the node involved in a shortest path routing in a cluster. Betweenness centrality gives high rating to those nodes that are likely to handle majority of traffic. Those nodes are said to be important for caching. However, high volume of traffic may cause space exhaustion in caching which leads to frequent cache replacement and hit rate reduction. Thus we consider a new hierarchical caching strategy based on node importance and content popularity, and a pairing between the two using a probability matrix. Our approach diversifies the use of nodes for caching, avoiding contents to be monotonically cached in certain nodes.

A. NODE IMPORTANCE DETERMINATION

Our node importance rating is based on the betweenness centrality of a node. Given a cluster network containing a set of nodes $\mathbf{U}$ where $\mathbf{U} \subset \mathbf{V}$, the betweenness centrality of node u can be calculated by each clusterhead according to the

following equation

$$B(u) = \sum_{s} \sum_{t \neq s} \frac{\delta_{s,t}(u)}{\delta_{s,t}} \tag{3}$$

where $s, t \in \mathbf{U}$ and $\delta_{s,t}(u)$ is the number of the shortest paths from node s to node t through node u. The quantity $\delta_{s,t}$ is the total number of the shortest paths from node s to node t [28].

All the members, gateways and clusterhead are rated based on betweenness centrality. Furthermore, the top p percentage of cluster nodes are classified as more important nodes, and the next p' percentage of cluster nodes are classified as less important nodes in this cluster. The remaining nodes are considered not important, and they are excluded from the list of important nodes. Clusterhead maintains a list of important nodes and the betweenness centrality rating for each node in the list. Let $N = |\mathbf{U}|$ which is the number of nodes in the cluster, the list contains $\lceil p \cdot N \rceil$ of more important and $\lceil p' \cdot N \rceil$ less important nodes.

B. CONTENT POPULARITY DETERMINATION

Another role of clusterhead is to collect and measure the content popularity in the Edge Cluster. Access routers which connect to content requesters are responsible for counting the request rate of contents and providing the information to their clusterhead. Based on the content popularity information reported by the access routers, clusterhead summarizes the content name and popularity in the cluster in descending order. Let P_H and P_L be the highest and lowest popularity scores respectively, and we subdivide the popularity within the range between P_H and P_L into K popularity classes.

Zipf-Mandelbrot distribution, which is a power-law distribution, is often used to model content popularity [16]. The interval of each class, I_p, is equal in log-scale, that is

$$I_p = \frac{\log P_H - \log P_L}{K} = \frac{\log \frac{P_H}{P_L}}{K}. \tag{4}$$

Each content belongs to a popularity class based on its popularity score. The highest popularity class is Class 1. Contents with popularity score of P_c satisfying the condition $\log P_c \geq \log P_H - I_p$ belong to class 1. The next class is Class 2. It groups contents with popularity of P_c satisfying $\log P_H - 2I_p \leq \log P_c < \log P_H - I_p$. Class 3 groups contents with popularity of P_c satisfying $\log P_H - 3I_p \leq \log P_c < \log P_H - 2I_p$, and so on. The last class is Class K. It carries the contents with popularity of P_c satisfying $\log P_c < \log P_L + I_p$.

In our hierarchical caching strategy, we consider that contents in the top q percentage of popularity classes are more popular. The remaining contents are less popular. The treatment of these two groups of contents will be different as we shall see in the next subsection.

C. PROBABILITY MATRIX COMPUTATION

In order to connect node importance and content popularity, a probability matrix, $\mathcal{P}$, is introduced to pair node importance and content popularity. $\mathcal{P}$ is a $(\lceil p * N \rceil + \lceil p' * N \rceil)$-by-$K$ matrix containing caching probabilities. Its element $a_{i,j}$ where $i \in [1, \lceil p * N \rceil + \lceil p' * N \rceil], j \in [1, K]$ describes the caching probability to be used by a node with importance rating of i in the Edge Cluster on a content with popularity class j. The paring of node importance and content popularity is indicated by a non-zero caching probability. Setting $a_{i,j} > 0$ instruct nodes with importance rating of i to cache content popularity class of j with a probability of $a_{i,j}$. Setting $a_{i,j} = 0$ simply disables the caching of content popularity class of j in the nodes with importance rating of i. The probability matrix, $\mathcal{P}$, is described in (5) and (6), as shown at the bottom of this page. The coefficients γ_1 and γ_2 in (6) are the weight factors where $\gamma_1 + \gamma_2 = 1$. Each element of probability matrix is calculated by the importance rating of i and popularity class of j in (6). Here the reciprocal function is used to assign higher probability for more important nodes and higher popularity class of contents.

Our design principle of $\mathcal{P}$ is that more important nodes should cache more popular contents and less popular contents should be only cached in less less important routers. Thus from (5), for non-zero $a_{i,j}$, we enforce that $a_{i,j} > a_{i,j+1}$ and $a_{i,j} > a_{i+1,j}$. The described cache policy has the same effect of caching more popular contents in the network edge

$$\mathcal{P} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,\lceil q \cdot K \rceil} & 0 & \cdots & 0 \\ a_{2,1} & a_{2,2} & \cdots & a_{2,\lceil q \cdot K \rceil} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{\lceil p \cdot N \rceil,1} & a_{\lceil p \cdot N \rceil,2} & \cdots & a_{\lceil p \cdot N \rceil,\lceil q \cdot K \rceil} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & a_{\lceil p \cdot N+1 \rceil,\lceil q \cdot K+1 \rceil} & \cdots & a_{\lceil p \cdot N+1 \rceil,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 0 & a_{\lceil p \cdot N \rceil + \lceil p' \cdot N \rceil,\lceil q \cdot K+1 \rceil} & \cdots & a_{\lceil p \cdot N \rceil + \lceil p' \cdot N \rceil,K} \end{bmatrix} \tag{5}$$

$$a_{i,j} = \begin{cases} 0 & 1 \leq i \leq \lceil p \cdot N \rceil, \lceil q \cdot K + 1 \rceil \leq j \leq K \\ 0 & \lceil p \cdot N + 1 \rceil \leq i \leq \lceil p \cdot N \rceil + \lceil p' \cdot N \rceil, 1 \leq j \leq \lceil q \cdot K \rceil \\ \gamma_1 \frac{1}{i} + \gamma_2 \frac{1}{j} & \text{otherwise} \end{cases} \tag{6}$$

similar to those in [21] and [29]. Nevertheless, our framework provides more flexible design for other effects or targets.

In the system, clusterhead collects the information of betweenness centrality and content popularity. After the collection, it calculates the probability matrix and sends the result as well as the corresponding class of content popularity to all important nodes in its cluster. Upon receiving the result from the clusterhead, all important nodes start performing caching using the probability matrix.

D. CONTENT CACHING

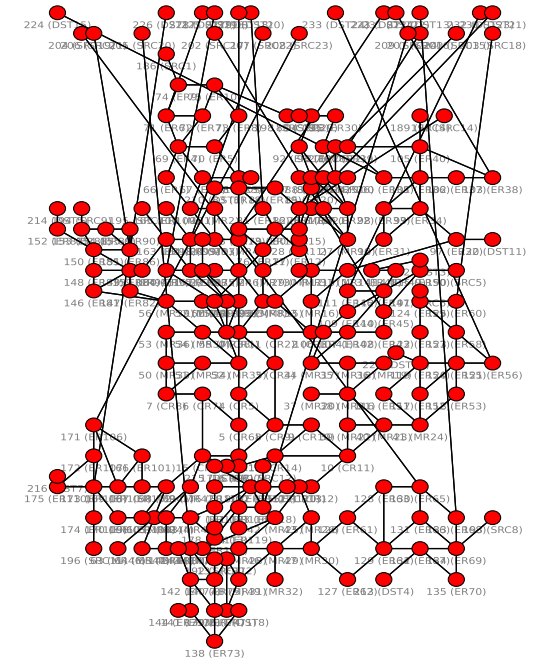
After receiving the probability matrix and popularity class list from the clusterhead, content caching operation for each important node is relatively straightforward. Content caching is generally based on probabilistic approach. Each important node first obtains the caching probability for each content popularity class. For each content, the node handles during routing operation, the content is cached with the specified caching probability in the node. Since the caching space is limited in each node, an important node may face caching storage exhaustion to further cache the received content. In this situation, a replacement strategy should be considered. We use a Least Recently Used (LRU)-Like strategy. When more space is needed, the node checks whether any cached content has a popularity class that is no longer specified in the latest content popularity class. LRU policy is used to discard these contents until adequate space is generated for the new content or all such contents are discarded. If further space is needed, LRU policy is used to all other contents.

Since content popularity changes from time to time, it is necessary for the clusterhead to periodically update the probability matrix and popularity class list. In our proposed solution, we enforce periodic update. Based on [30], an update interval of 5 to 60 minutes is considerable appropriate.

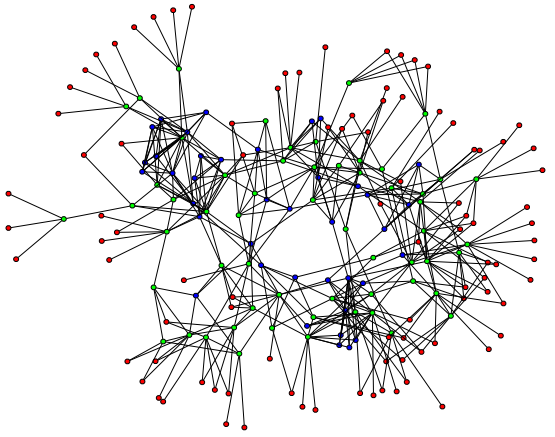
E. CLUSTER-BASED ROUTING

To complete our solution, we further need a cluster-based routing protocol to efficiently find cached and non-cached contents and deliver them over the clustered network. If the content is not cached, the request will be forwarded to the content provider directly according to the shortest path routing algorithm. Some candidates such as those described in [31], [32] can be used for this purpose. We use Open Shortest Path First for Named-data (OSPFN) which is a default routing policy of NDN.

When clusterhead receives an Interest, it will check the popularity class of the requested content. If its popularity class maps with more important or less important nodes, the clusterhead will floods the Interest to these nodes in its cluster accordingly. Each node receiving the request checks its Content Store [33] and responds to the request directly. Thus, only clusterhead implements flooding strategy in a limited range. Methods such as those presented in [34] and [35] provide efficient lookup for cached contents in a network may be used.



(a)



(b)

FIGURE 2. The topology of simulation. (a) The topology of GEANT-based network. (b) The topology of AS3967 network.

V. EVALUATION AND ANALYSIS

We use ndnSIM [36] simulation to evaluate the performance of our hierarchical cluster-based caching. We build two simulation scenarios with different network topologies. The first scenario is based on the GEANT network topology [37]. The simulated network topology is given in Fig. 2(a). In the network, all routers in Core Layer are chosen from the nodes with more than 100 Gbps bandwidth in the GEANT network. The round trip time (RTT) between any two nodes in the core is measured by the HADES system [38]. We set the node limitation for Edge Cluster σ to 14. The RTT between two cluster nodes in Edge Cluster is a random value uniformly distributed between 5 ms and 40 ms. Node degree of Edge Cluster is also chosen by random algorithm from 2 to 6 [39].

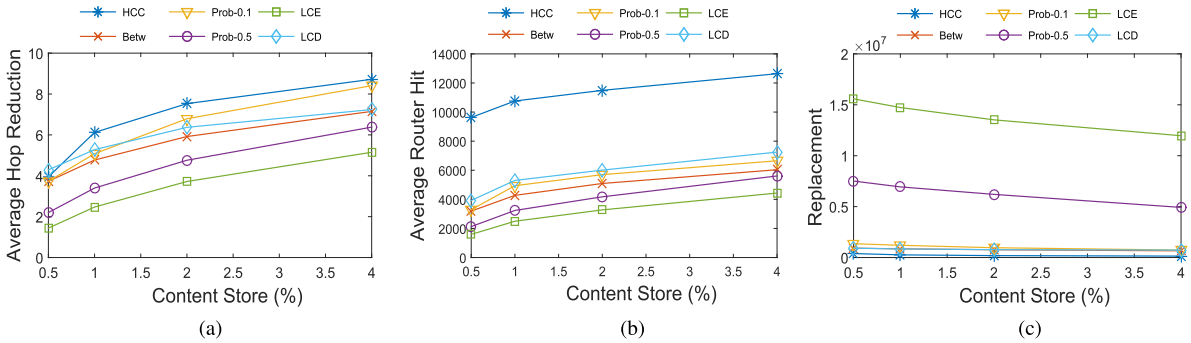


FIGURE 3. In the GEANT-based network, impact of cache storage of different strategies. (a) Average hop reduction. (b) Average router hit. (c) Replacement.

TABLE 1. Nodes of simulation topology.

Parameters \ Topology	GEANT-based	AS3967
Core Layer nodes	18	39
Edge Layer nodes	168	153
Edge Clusters	12	10
percentage of important nodes p and p'	30% and 10%	
percentage of popularity class q	60%	
proportion of Content Store	0.5%, 1%, 2%, 4%	
parameter of Zipf distribution	0.6, 0.85, 1, 1.2	
required contents	$1 * 10^4$	
requesters and providers	(12, 12)	

The second scenario follows Exodus AS-3967 network. Link bandwidth, routing metric, link delay are all based on Rocketfuel traces [40]. As shown in Fig. 2(b), the topology consists of 192 nodes including 39 backbone nodes, 58 gateway nodes and 95 leaf nodes. In AS-3967 network, node limitation of Edge Cluster σ is set to 16.

The settings of main parameters used in our simulation are shown in Table 1. The request arrival rate at each content requester follows Poisson arrival process. The simulation time is set to 1000s and the Interest packet generation rate is set to 100 packets per second. Similar to [17], all simulations go through a warm-up phase where the content popularity, node importance and probability matrix calculation are completed and the information is distributed.

Three metrics are introduced for evaluating the performance of caching strategies. They are

- Average Hop Reduction: when contents can be retrieved from a node, the number of hops involved in packet routing for content retrieval is reduced. This metric measures the average number of hops reduced for the communication between the requesters and providers due to caching.
- Average Router Hit: the average number of Interests served by each router. This metric shows the efficiency of caching.
- Replacement: the frequency of old contents in routers being replaced by new ones, indicating the caching overhead.

Compared with the performance of our HCC, we select four other caching strategies as follows. LCE [33] is the basic scheme in NDN that caches contents everywhere. In LCD [13], contents are cached in the downstream router. Betw is a betweenness centrality-based caching scheme [17] which caches contents in the router with the highest betweenness centrality. Prob(p) uses a fixed probability to decide whether a content should be cached [13]. In our simulation, we tested both 0.5 and 0.1 probability values for Prob(p).

A. IMPACT OF CONTENT STORAGE

In this subsection, we study the impact of cache storage in each node varying from 50 to 400 contents on the caching performance. There are 12 content requesters and 12 content providers in both topologies. Each content provider is set to store 10^4 contents. In other words, a cache storage of 100 contents in a node corresponds to 1% of contents hosted in a content provider. We study the performance of caching with sizes of cache storage in each node setting to 0.5%, 1%, 2% and 4%. Content popularity follows the Zipf-Mandelbrot distribution with skewness parameter $\alpha = 0.85$ [16]. We report simulation results in Fig. 3 and Fig. 4. As can be seen, our proposed HCC strategy shows clear performance advantages compared with its peers.

We first study LCE which is a scheme to blindly leave copies everywhere. From Fig. 3(c) and Fig. 4(c), it is not surprised to see that its replacement rate is high because caching is done pervasively without evaluating their popularity. The high replacement rate contributes to the low router hit rate as shown in Fig. 3(b) and Fig. 4(b). With many contents eventually retrieving from the providers, the contents are delivered through more hops in the network. The average hop reduction due to caching as shown in Fig. 3(a) and Fig. 4(a) for LCE is thus low.

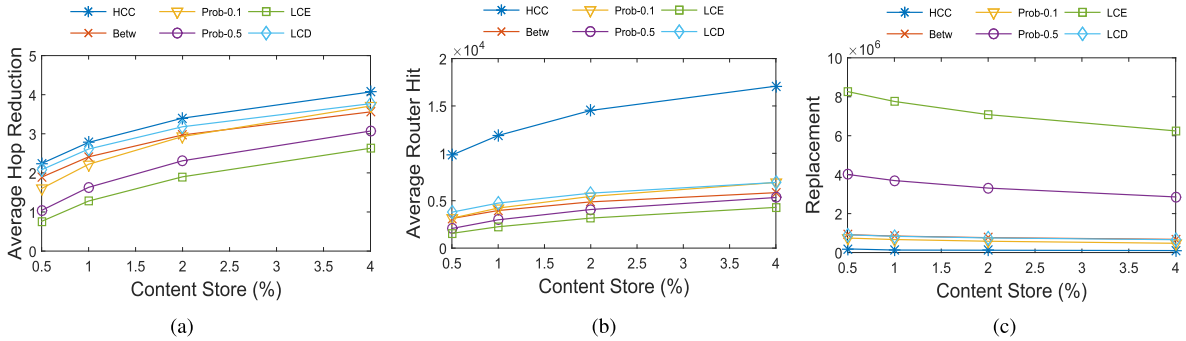


FIGURE 4. In the AS3967 network, impact of cache storage of different strategies. (a) Average hop reduction. (b) Average router hit. (c) Replacement.

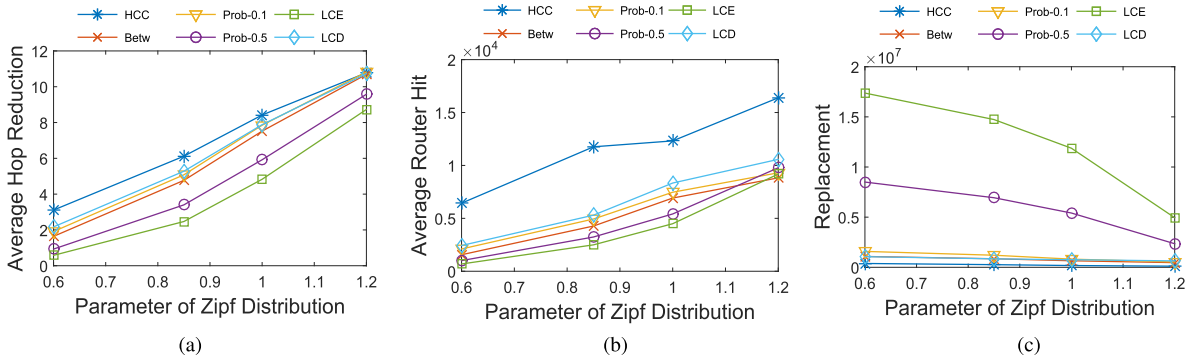


FIGURE 5. In the GEANT-based network, impact of content popularity distribution of different strategies. (a) Average hop reduction. (b) Average router hit. (c) Replacement.

Prob(p) with 0.5 and 0.1 probability settings respectively cache 50% and 10% of the contents in the routers. They reduce replacement rate, increases the hit rates as well as hop reduction. We see that lower probability setting generally gives better performance. However, optimal setting of the probability depends on many dynamic factors which is difficult to determine in real time.

LCD caches contents in the downstream router. By making use of the location for content caching, its replacement rate is low, router hit rate and average hop reduction are comparatively high. The results show the advantage of including location information for content caching. Betw uses betweenness centrality-based caching scheme which explicitly consider network topology for caching. It has similar performance compared with LCD.

By coordinating network topology and content popularity for caching in our proposed solution, we see clear performance advantages compared with the others. Particularly for average router hit shown in Fig. 3(b) and Fig. 4(b), a clear gap is seen between our solution and all other schemes.

B. IMPACT OF CONTENT POPULARITY DISTRIBUTION

It is demonstrated that content popularity distribution generally follows a Zipf-like distribution [41]. Different applications and scenarios may exhibit different values of skewness parameter. In general, the value is linked to the user request behavior where a higher value of skewness parameter indicates that requests are more concentrated on some specific contents. In this subsection, we study the impact of different skewness values on the caching performance. Fig. 5 and Fig. 6 shows the performances of all caching strategies with skewness values of Zipf distribution ranging from 0.6 to 1.2. The range setting is based on the study given in [42]. According to [43], content popularity of a user generated content (UGC) service follows Zipf distribution with skewness value of approximately 0.88. For a video on demand (VOD) service, statistics in China shows a value between 0.65 and 1 [44]. Our simulation also considers 12 content requesters and 12 content providers with 10^4 contents each, and cache storage of 100 in each node.

The average hop reduction of all the strategies reported in Fig. 5(a) and Fig. 6(a) increases as the skewness value increases. This is because with higher skewness value, a smaller set of contents are access more frequently which promotes the effectiveness of caching. Among all strategies, our proposed HCC strategy consistently records the highest reduction.

Performance advantages are also shown for average router hit and replacement for our proposed HCC strategy in both Figs. 5(b)-5(c) and Figs. 6(b)-6(c). Again, the average router

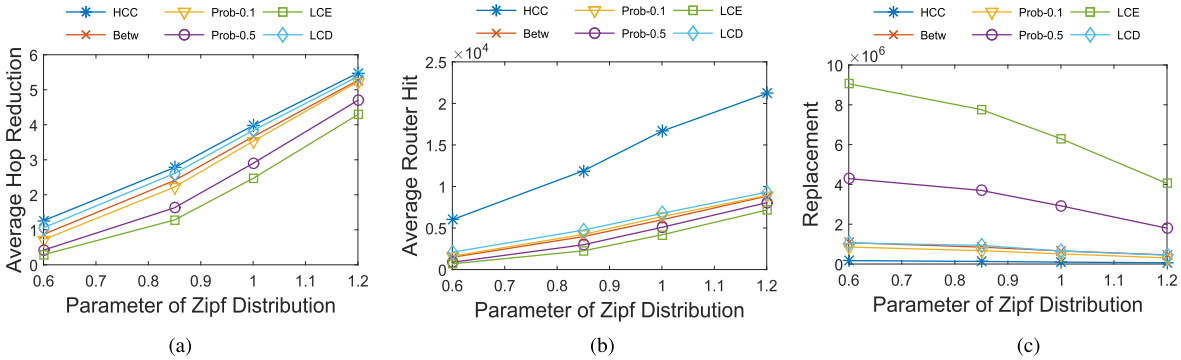


FIGURE 6. In the AS3967 network, impact of content popularity distribution of different strategies. (a) Average hop reduction. (b) Average router hit. (c) Replacement.

hit, a gap between our proposed strategy and all other strategies is clearly shown.

C. ANALYSIS OF COMMUNICATION OVERHEAD

The communication overhead of HCC is embodied in both initialization phase and update phase. In initialization phase, the advertisement overhead between clusterheads and members during cluster formation is $(\sigma+1+\eta)(V_O-V_C)$ according to [6], where σ is node limitation, η is number of clusters, V_O is the whole number of nodes, V_C is the number of Core Layer nodes. And the number of messages for collecting and distributing probability matrix by clusterheads is $3\cdot\sigma\cdot\eta$. And in content popularity update phase, the advertisement overhead is $\lfloor T_s/f\rfloor\cdot(2\sigma)\cdot\eta$, where T_S is simulation time, f is update interval of popularity. Thus the total communication overhead is shown as (7).

$$C_N=(\sigma+1+\eta)\cdot(V_o-V_C)+(3+2\lfloor T_s/f\rfloor\cdot\sigma\cdot\eta \qquad (7)$$

Take AS-3967 as an example, each content provider stores 10^4 contents, the size of cache storage in each node is set to 100. Based on [30] and [45], when the update interval f is set to 30 min, the ratio between communication overhead of HCC and the total packets transmitted is less than 0.19%.

Compared with other 4 caching strategies, they don't need message interaction because all the nodes implement the caching decision individually. Even though HCC has a little communication overhead, it obtains obvious performance improvement.

VI. CONCLUSION

In-network caching offers prompt response and ease traffic load in the core network. Since in-network caching operates in a distributed environment, challenges exist in designing an efficient in-network caching for a large-scale network. In this paper, we introduced a two-layer Hierarchical Cluster-based Caching strategy. Our strategy jointly considers both the location of the node and the popularity of content to make caching decision. To overcome high overhead, we used cluster-based approach to scale a network into multiple autonomous groups for better manageability. A centralized decision on caching policy was made in the clusterhead, which was eventually described by a probability matrix. Based on the probability matrix, each router made its caching decision individually. We implemented our proposed strategy in NS3 based on ndnSIM project. Our tests showed that in both GEANT-based network and AS3967 network, our strategy consistently outperforms other strategies in several aspects. In particular, our strategy gives the highest average hop reduction for content retrieval showing that not only more contents are retrieved from the cache, but also the cached contents are nearer to the requesters. This result confirms the benefit of jointly considering location and content popularity for in-network caching.

In the future work, we will consider to optimize the updating algorithmic of content popularity, by using flexible update interval of sliding window instead of static update period, which makes caching decision more accurately.

REFERENCES

[1] *Named Data Networking*. [Online]. Available: http://www.named-data.net
[2] 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.
[3] M. Amadeo *et al.*, "Information-centric networking for the Internet of things: Challenges and opportunities," *IEEE Netw. Mag.*, vol. 30, no. 2, pp. 92–100, Mar. 2016.
[4] S. Wang, J. Bi, J. Wu, and A. V. Vasilakos, "CPHR: In-network caching for information-centric networking with partitioning and hash-routing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2742–2755, Oct. 2016.
[5] J. A. Torkestani and M. R. Meybodi, "A mobility-based cluster formation algorithm for wireless mobile ad-hoc networks," *Cluster Comput.*, vol. 14, no. 4, pp. 311–324, 2011.
[6] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks," *IEEE Commun. Surveys Tuts.*, vol. 7, no. 1, pp. 32–48, 1st Quart., 2005.
[7] H. J. Choe, P. Ghosh, and S. K. Das, "QoS-aware data reporting control in cluster-based wireless sensor networks," *Comput. Commun.*, vol. 33, no. 11, pp. 1244–1254, 2010.
[8] G. Chen, C. Li, M. Ye, and J. Wu, "An unequal cluster-based routing protocol in wireless sensor networks," *Wireless Netw.*, vol. 15, no. 2, pp. 193–207, 2009.
[9] A. Daeinabi, A. G. Pour Rahbar, and A. Khademzadeh, "VWCA: An efficient clustering algorithm in vehicular ad hoc networks," *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 207–222, 2011.

[10] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," Dept. Comput. Sci., Univ. California, Oakland, CA, USA, Tech. Rep. NDN-0005, 2012.
[11] H. Yan, D. Gao, and W. Su, "A hierarchical cluster-based caching for named data networking," in *Proc. IEEE/CIC Int. Conf. Commun. China*, Jul. 2016, pp. 1–6.
[12] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system," in *Proc. 7th ACM SIGCOMM Conf. Internet Meas.*, 2007, pp. 1–14.
[13] 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, 2006.
[14] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proc. 2nd Ed. ICN Workshop Inform.-Centric Netw.*, 2012, pp. 55–60.
[15] Y. Wang, M. Xu, and Z. Feng, "Hop-based probabilistic caching for information-centric networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 2102–2107.
[16] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "Wave: Popularity-based and collaborative in-network caching for content-oriented networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Mar. 2012, pp. 316–321.
[17] 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, 2013.
[18] J. Li *et al.*, "Popularity-driven coordinated caching in named data networking," in *Proc. 8th ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, Oct. 2012, pp. 15–26.
[19] W. Wang *et al.*, "CRCache: Exploiting the correlation between content popularity and network topology information for ICN caching," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2014, pp. 3191–3196.
[20] J. Ren *et al.*, "MAGIC: A distributed max-gain in-network caching strategy in information-centric networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2014, pp. 470–475.
[21] Z. Ming, M. Xu, and D. Wang, "Age-based cooperative caching in information-centric networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Mar. 2012, pp. 268–273.
[22] J. M. Wang, J. Zhang, and B. Bensaou, "Intra-AS cooperative caching for content-centric networks," in *Proc. 3rd ACM SIGCOMM Workshop Inform.-Centric Netw.*, 2013, pp. 61–66.
[23] J. Ji, M. Xu, and Y. Yang, "Content-hierarchical intra-domain cooperative caching for information-centric networks," in *Proc. 8th Int. Conf. Future Internet Technol.*, 2014, p. 6.
[24] J. Dai, Z. Hu, B. Li, J. Liu, and B. Li, "Collaborative hierarchical caching with dynamic request routing for massive content distribution," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 2444–2452.
[25] L. Li, D. Alderson, W. Willinger, and J. Doyle, "A first-principles approach to understanding the Internet's router-level topology," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 3–14, 2004.
[26] M. Chatterjee, S. K. Das, and D. Turgut, "An on-demand weighted clustering algorithm (WCA) for ad hoc networks," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, vol. 3. Nov. 2000, pp. 1697–1701.
[27] D. Rossi and G. Rossini, "On sizing CCN content stores by exploiting topological information," in *Proc. IEEE 31st Int. Conf. Comput. Commun.*, Mar. 2012, pp. 280–285.
[28] H. Wang, J. Martin Hernandez, and P. Van Mieghem, "Betweenness centrality in a weighted network," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 77, p. 046105, Apr. 2008.
[29] J. M. Wang and B. Bensaou, "Progressive caching in CCN," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 2727–2732.
[30] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: A scalable wide-area Web cache sharing protocol," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 281–293, Jun. 2000.
[31] L. Wang, A. K. M. M. Hoque, C. Yi, A. Alyyan, and B. Zhang, "OSPFN: An OSPF based routing protocol for named data networking," Dept. Comput. Sci., Univ. Memphis, Memphis, TN, USA, Tech. Rep. NDN-0003, 2012.
[32] A. K. M. M. Hoque, S. O. Amin, A. Alyyan, B. Zhang, L. Zhang, and L. Wang, "NLSR: Named-data link state routing protocol," in *Proc. 3rd ACM SIGCOMM Workshop Inform.-Centric Netw.*, 2013, pp. 15–20.
[33] 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. Experim. Technol.*, Rome, Italy, Dec. 2009, pp. 1–12.
[34] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 3, pp. 62–67, 2012.
[35] R. Chiocchetti, D. Perino, G. Carofiglio, D. Rossi, and G. Rossini, "Inform: A dynamic interest forwarding mechanism for information centric networking," in *Proc. 3rd ACM SIGCOMM Workshop Inform.-Centric Netw.*, 2013, pp. 9–14.
[36] *NS-3 Based Named Data Networking (NDN) Simulator*. [Online]. Available: http://ndnsim.net/2.0
[37] *GEANT Project Website*. [Online]. Available: http://www.geant.net/
[38] *DFN-LABOR Website*. [Online]. Available: http://www.win-labor.dfn.de/English/service_sidemap.html
[39] H. Haddadi *et al.*, "Beyond node degree: Evaluating AS topology models," Dept. Comput. Lab., Univ. Cambridge, Cambridge, U.K., Tech. Rep. UCAM-CL-TR-725, Jul. 2008. [Online]. Available: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-725.pdf
[40] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 4, pp. 133–145, 2002.
[41] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *Proc. 18th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, vol. 1. Mar. 1999, pp. 126–134.
[42] C. Fricker, P. Robert, J. Roberts, and N. Sbihi, "Impact of traffic mix on caching performance in a content-centric network," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Mar. 2012, pp. 310–315.
[43] Y. Carlinet, B. Kauffmann, P. Olivier, and A. Simonian, "Trace-based analysis for caching multimedia services," Dept. Orange Labs, France Telecom., Paris, France, Tech. Rep., 2011.
[44] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng, "Understanding user behavior in large-scale video-on-demand systems," *ACM SIGOPS Oper. Syst. Rev.*, vol. 40, no. 4, pp. 333–344, 2006.
[45] H. Dai, Y. Wang, H. Wu, J. Lu, and B. Liu, "Towards line-speed and accurate on-line popularity monitoring on NDN routers," in *Proc. Int. Symp. Quality Ser.*, May 2014, pp. 178–187.

HUAN YAN received the Ph.D. degree in communications and information system from Beijing Jiaotong University, Beijing, China, in 2016. Her research interests are information-centric network, next generation network architecture, and transport layer protocol optimization.

DEYUN GAO received the B.Eng. and M.Eng. degrees in electrical engineering and the Ph.D. degree in computer science from Tianjin University, China, in 1994, 1999, and 2002, respectively. He was a Research Associate with the Department of Electrical and Electronic Engineering, The Hong Kong University of Science and Technology, Kowloon, for one year. He then spent three years as a Research Fellow with the School of Computer Engineering, Nanyang Technological University, Singapore. In 2007, he joined the faculty at Beijing Jiaotong University as an Associate Professor of the School of Electronics and Information Engineering and became a Full Professor in 2012. In 2014, he was a Visiting Scholar with the University of California at Berkeley, USA. His research interests are Internet of Things, vehicular networks, and next-generation Internet.

WEI SU received the Ph.D. degree in communication and information system from Beijing Jiaotong University, China, in 2008. From then, he was with the School of Electronics and Information Engineering, Beijing Jiaotong University. In 2015, he was a Visiting Scholar with Future University, Hakodate, Japan. He is currently a Full Professor with Beijing Jiaotong University. His research interests include future Internet architecture, network security, and mobility management.

CHUAN HENG FOH (S'00–A'02–M'03–SM'09) received the M.Sc. degree from Monash University, Australia, in 1999, and the Ph.D. degree from the University of Melbourne, Australia, in 2002. He was a Lecturer with Monash University for six months. In 2002, he joined Nanyang Technological University, Singapore, as an Assistant Professor until 2012. He is currently a Senior Lecturer with the University of Surrey. He has authored or co-authored over 100 refereed papers in international journals and conferences. His research interests include protocol design and the performance analysis of various computer networks, including wireless local area and mesh networks, mobile ad hoc and sensor networks, Internet of Things, 5G networks, and data center networks. He is the Vice-Chair (Europe/Africa) the IEEE Technical Committee on Green Communications and Computing. He is currently an Associate Editor of the IEEE Access, the IEEE Wireless Communications, and the *International Journal of Communications Systems*.

HONGKE ZHANG received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 1992. He is currently a Professor with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing, China, where he is also directs the National Engineering Laboratory on Internet Technology for Next Generation Internet in China. His research has resulted in many research papers, books, patents, systems, and equipment in the areas of communications and computer networks. He has served on the editorial boards of several international journals.

ATHANASIOS V. VASILAKOS received the Ph.D. degree in computer engineering from the University of Patras, Patras, Greece, in 1988. He is currently a Professor with the Luleå University of Technology, Sweden. He served or is an Editor or Guest Editor for many technical journals, such as the IEEE Transactions on Network and Service Management, the IEEE Transactions on Cloud Computing, the IEEE Transactions on Information Forensics and Security, the IEEE Transactions on Cybernetics, the IEEE Transactions on Nanobioscience, the IEEE Transactions on Information Technology in Biomedicine, the *ACM Transactions on Autonomous and Adaptive Systems*, and the IEEE Journal on Selected Areas in Communications.