A Popularity-Based Collaborative Caching Algorithm for Content-Centric Networking

Xiaodong Zhu[1,2], Weining Qi[1], Jinlin Wang[1], and Yiqiang Sheng[1]
[1]National Network New Media Engineering Research Center, Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, China
[2]University of Chinese Academy of Sciences, Beijing 100190, China
Email: {zhuxd, qiwn, wangjl, shengyq}@dsp.ac.cn

Abstract —The key aspect of Content-Centric Networking (CCN) is that the contents are cached by in-network nodes, and hence the caching scheme is important for the efficient use of cache and content delivery. Although the caching scheme in CCN has been widely researched, it is still a great challenge to better utilize in-network caching of CCN with an appropriate caching scheme. In this paper, we propose an effective popularity-based collaborative caching algorithm with the idea to reduce content redundancy within network and make popular contents near clients. The algorithm is lightweight and implicit collaborative with real-time analysis local popularity and little change to the pipeline of CCN. We also analyze that the proposed algorithm is suitable for solving the lack of contents and high external bandwidth charge problems of China Mobile Communications Corporation. The algorithm is simulated with many factors that may have impacts on the performance, and our algorithm is effective, scalable and stable for various topologies, popularity estimate methods and features. The results of simulation show that our algorithm outperforms the existing classical algorithm in access hops and server hit proportion.

Index Terms—Content-Centric networking, China mobile network, popularity-based, collaborative caching

I. INTRODUCTION

Information-Centric Networking (ICN) [1] has aroused great attention with the idea to design a new network architecture in a revolutionary rather than evolutionary way. ICN uses named data as the core of the tiny-waist network architecture, which is different from the traditional IP network [2]. Such information-centric design can address the problem brought by the changing application schema, which focuses on the information rather than the host providing information. Video traffic has occupied a major part of the Internet bandwidth [3], [4], which is a typical change for the Internet application schema. And several Information-Centric Networking architectures have been proposed, such as TRIAD [5], DONA [6], COMET [7], SCAFFOLD [8], and Content-Centric Networking (CCN) [9], [10]. Additionally, CCN has attracted a lot of attention quickly since it was proposed in 2009. The NDN project [11], which is led by the University of California at Los Angeles and funded by the U.S. National Science Foundation under its Future Internet Architecture Program, is based on the design of CCN. Caching contents in network is CCN's key feature, and in-network cache can efficiently use cache and deliver contents.

The communication of CCN is based on the content, and each CCN node has a certain storage capacity. Content Store (CS), Pending Interest Table (PIT) and Forwarding Information Base (FIB) are three key components of the CCN router. Consumers will send Interest packets to request contents. When the Interest packet gets the request content in a node's content store, the response content will get back along the way the Interest packet comes. Network caching technology plays a major role in CCN, and the caching strategy has a significant impact on the performance of CCN. Caching replacement algorithm is from web caching [12] and has been deeply researched, such as LRU and LFU. But previous algorithms just focus on the single node and sophisticated cache replacement algorithm is not appropriate for the requirements for wire-speed execution of ICN. Rossini *et al*. [13] presented cache replacement algorithm is not critical for performance improvement. Network caching decision strategies of CCN derive from web caching, CDN and P2P systems. Although we get lots of references for caching decision algorithm from above systems, there are still significant differences between caching decision algorithms in CCN with them. And more efforts are needed to research caching decision in CCN to better utilize in-network caching. The main objectives of in-network caching are: (a) deliver the content and a content will be got faster if it is cached along the way from the client to the server, (b) reduce the traffic as contents are transmitted less when cache hit happens, (c) relieve server load since requests can get response from in-network nodes. Thus, these are also the main objectives of caching decision research to better utilize in-network caching.

China Mobile Communications Corporation (CMCC) is one of the main providers of fixed network business in China. And the quality of service is a focal factor for CMCC to compete with China Telecom and China Unicom. While lack of contents of CMCC is the major

Manuscript received July 1, 2016; revised October 17, 2016.
This work was supported by the Special Fund for Strategic Pilot Technology Chinese Academy of Sciences (No. XDA06010302).
Corresponding author email: wangjl@dsp.ac.cn.
doi:10.12720/jcm.11.10.886-895

constraint and the external bandwidth charge is pretty high. Thus a towering performance ICN architecture with appropriate caching decision algorithm is the urgent needs of CMCC. The lack of contents and high external bandwidth charge problems of CMCC will be analyzed in this paper. We will also abstract the topology of CMCC to use in our experiment to better reflect our algorithm.

In this paper, we provide a novel caching algorithm to efficiently distribute contents and use cache to get better performance for CCN network. Firstly, we present a collaborative caching algorithm based on content popularity. The algorithm hierarchically and accurately locates contents in network. The algorithm is lightweight and implicit collaborative with real-time analysis local popularity and little change to the pipeline of CCN. Secondly, we prove the advantage of local popularity based on real-time analysis compared to the global popularity which needs additional message exchange and change to the CCN. Finally, we demonstrate that our algorithm is suitable for solving the lack of contents and high external bandwidth charge problems of CMCC.

The rest of the paper is organized as follows. The related work is surveyed in Section II. Section III describes our novel algorithm. In Section IV, the fixed network of CMCC is discussed. In Section V, the experiments are discussed in detail. Finally, conclusions and future work are addressed in Section VI.

II. Related Work

Caching decision, which decides whether caching a passing content, is very important for the performance of CCN network. Dabirmoghaddam et al. [14] presented that caching at the edge of the network outperforms the traditional on-path, while the simulation is in an unstructured network with arbitrary topology. Tyson et al. [15] proposed caching at the edge can improve performance but pervasive cache can get better improvement. Leave copy everywhere (LCE) [16] is the most simple caching decision strategy, and it is also the default cache decision strategy of CCN. Content is cached by all routers along the path in LCE. This leads to redundant cache in network and reduces the diversity of cache contents. And content with high popularity cannot stay long in a network with high speed content replacement. Therefor, reducing redundant contents and discriminating against unpopular contents is important for caching decision research to deliver the content, reduce the traffic and relieve server load.

Leave Copy Down (LCD) [17], [18] distributes the contents based on the popularity. Contents are only cached in the next hop of the cache hit node in LCD. But it will take much time for the contents to come near consumers with a lot of cache hits and also leave redundant cache along the path. WAVE [19] adjusts the number of chunks caching in each node based on the popularity of the document. WAVE uses the document as a unit to statistic the number of access. It considers the relevance of chunks compared to LCD. Xu *et al*. [20] proposed a utility-based caching mechanism. The utilities of the contents are tracked by nodes for making the caching decisions. Age-based Cooperative Caching [21] proposes Age-based mechanism to distribute popular contents. A cached content will be replaced if age of it is expired when a new content arrives. And more popular contents are expected to be cached near clients in the stable stage. These popular-based algorithms can make popular contents at the edge, but still remaining redundant caches. Prob-PD [22] makes use of both the popularity and node location to expand popular contents. Contents with higher popularity have the higher probability to be cached at the edge of the network. While Prob-PD only reduces the redundancy of content to a certain extent. Li *et al*. [23] also proposes a popularity-driven algorithm, while additional message exchange is needed for measuring the popularity and these messages are not suitable for the CCN process pipeline. Additionally, the mechanism of the algorithm is too complex.

As discussed above, the existing algorithms still can't accurately place contents in network or reduce redundancy contents with a lightweight mechanism. In order to reduce redundant contents and make popular content near clients, we propose a popularity-based collaborative caching algorithm. The algorithm hierarchically and accurately locates contents in network. And it avoids duplicating content on path. The mechanism of the algorithm is lightweight with real-time analysis local popularity and little change to the pipeline of CCN.

III. Collaborative Caching Algorithm

In this section, a novel collaborative caching algorithm is described in detail. Firstly, the packet format used in our algorithm is shown in Fig. 1. It needs only minor changes to the current CCN packet format and can support high requirement for the processing rate. Secondly, specific details of the collaborative storage algorithm are discussed. At last, we present the way to measure the popularity and analyze the difference between local popularity and global popularity in our algorithm.

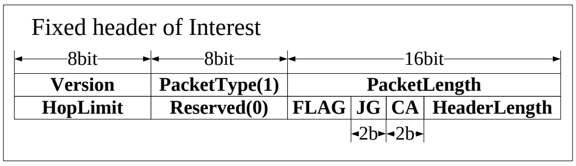


Fig. 1. Interest packet format

A. Packet Format

A CCNx [24] packet includes a fixed header and optional headers in TLV format and payloads. The fixed header carries main informations of CCN and can support fast processing with the need of wire-speed execution.

As shown in Fig. 1, we use 4 bits of the flag space of the Interest packet's fixed header to support our collaborative storage algorithm. The JG tag is used to determine whether or not to execute the popularity compare process. We use one for true, and zero for false. And the default value is true. If the value of the JG tag is one, the step of popularity compare in the algorithm will be executed. And the CA tag is used to determine whether or not to cache the content. The value of the CA tag ranges from zero to two. The default value is two. The returned content will not be cached by a router if the matched Interest packet's CA tag is zero.

B. *Collaborative Caching Algorithm*

In this section, we propose a collaborative caching algorithm that can support popularity based caching with little change to the CCN process pipeline. Our main objective of the algorithm is to avoid redundant content within network and cache more popular contents near the consumers.

Algorithm 1 Interest process algorithm

Input: Interest, CS
Precondition: receive a Interest packet with no response content in CS
if CS is full **then**
  check JG tag
  if JG tag is true **then**
    compare the popularity with cached data
    if Interest's popularity is higher **then**
      set JG tag false and set CA tag 1
    else
      set CA tag 0
    endif
  endif
endif

Algorithm 2 Forward Interest algorithm

Input: Interest
Precondition: prepare to forward Interest
Check CA tag
if CA $tag = 1$
  CA $tag = tag - 1$
endif

Algorithm 3 Content process algorithm

Input: content, PIT
Precondition: content matches PIT
check CA tag
if CA $tag != 0$
  store this content in CS
else
  don't store the content
endif

As shown in Algorithms 1, when the CS is not full at the beginning, Interest packets are processed as the original way and all contents will be cached by the routers along the way. A router will first check the JG tag to determine whether or not to execute the popularity compare process. This will happen when the CS is full and can not match new Interest. If the JG tag is true, the router will compare the popularity of the request content with cached data. If the Interest's popularity is higher, it means the returned content for this Interest should be cached here and need not be cached by other routers. Then, the router will set the JG tag false and the CA tag as one. If the Interest's popularity is lower, it means the router doesn't need to cache and will leave other routers along the way to determine whether or not to cache. As a result, the router will just set CA tag as zero. Then Interest will be processed by PIT and FIB just as usual.

As shown in Algorithms 2, the CA tag of the Interest will be checked before the Interest is forwarded. If the CA tag is one, just minus one to make it zero.

As shown in Algorithms 3, when a router gets a content matches PIT, the CA tag of the Interest fitting this content in PIT will be checked before caching the content. The content will be cached only if the CA tag is not zero.

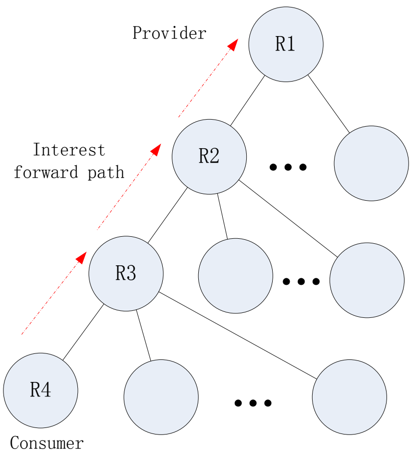


Fig. 2. An example of the algorithm

An example of the collaborative caching algorithm is shown in Fig. 2. The initial value of the CA tag is two and the JG tag is true. We assume content A has the lowest popularity, content B is the most popular, and content C is less popular than content B. And each router can cache one content. The value of the JG tag and the CA tag for each Interest in each router after the process can be seen in Table I. At the beginning, a consumer sends an Interest requesting content A from the producer. With empty CS, Routers along the path will do no extra processes and the returned content A will be cached by router R1, R2, R3, R4 along the way with the default CA tag equals two. CS is full now, and the consumer sends an Interest requesting content B. The Interest firstly comes to router R4 and gets no cache hit. After popularity compare, R4 assumes the returned content B should be cached here as content B is more popular than content A which is cached in R4. The Interest gets no cache hit at R3, R2 and R1, and finally gets content B from the producer. While R1, R2 and R3 will not execute the popularity compare process as R4 has decided to cache and set the JG tag of the Interest as false. R1, R2, R3 will not cache content B as the CA tag of the Interest has been changed to zero by R4. And content B will be cached by R4 with CA tag equals one. Then content A is replaced by content B in R4. Thus content B with higher popularity will be cached at the router near the consumer and there is no other copy along the way. Then the consumer sends an Interest requesting content C. The Interest also firstly arrives at router R4 and gets no cache hit. After popularity compare, R4 assumes the returned content C should not be cached here as content C is less popular than content B. The CA tag of the Interest will be set as zero and no change to the JG tag, which will make R4 not cache the returned content C and let later routers make decision. The Interest gets no cache hit at R3 either, and R3 will compare popularity as the JG tag is still true. Content C is more popular than content A, so R3 assumes the returned content C should be cached here. The Interest gets no cache at R1, R2 and gets content from the producer. Finally, content C with lower popularity is only cached by R3 along the way. Content B which is the most popular is cached by R4. Content A is in R2 and R1. The contents with different popularity are hierarchically and accurately located in network.

TABLE I: VALUE OF JG/CA TAG IN EACH ROUTER

Interest	R1	R2	R3	R4
For A	True/2	Ture/2	True/2	True/2
For B	False/0	False/0	False/0	False/1
For C	False/0	False/0	False/1	Ture/0

C. Popularity Estimate

The content's popularity can be represented as P_{ct}, which is defined as the requests' number for the content divided by the total. We can count requests in a moving window, and update the window by the exponentially weighted moving average algorithm. Additionally, requests can simply be counted since counts of the router are initialized. We use the latter method to estimate the popularity in our simulation. For a router, the number of the request for the content ct can be represented as N_{ct}, and $\sum_{ct'=1}^{CT} N_{ct'}$ means the total number of requests for all kinds of contents. So the popularity P_{ct} can be defined as in (1):

$$P_{ct} = \frac{N_{ct}}{\sum_{ct'=1}^{CT} N_{ct'}} \tag{1}$$

The popularity of the content can also be defined by the proportion of the requests for this content of the total requests generated by all consumers. Obviously, global popularity can more accurately reflect the true status of the network traffic than the above local popularity. But it will need much additional interactive information for each router to know the global popularity of the content. Moreover, if we use global popularity to compare, the most popular content will be cached by intermediate nodes when the content store isn't full. And such content can't be replaced without expiration checking mechanism as its global popularity is higher. While such kind of contents may not be hit because contents with the highest popularity have been cached at the edge of the network with our collaborative storage algorithm. This can result in data redundancy within network. Such situation can be avoided if we use the local popularity to compare in our algorithm. For routers that are at the edge of the network, local popularity can reflect the global popularity after a long time statistics. The highest global popularity contents can still be cached at the edge of the network. And for intermediate routers, secondary highest popularity contents can get higher local popularity and be cached. The diversity of cache within network can be enhanced. Specific performance analysis will be discussed in Section V.

IV. CHINA MOBILE NETWORK

In this section, we first describe the topology of CMCC network. And then we discuss the lack of contents and high external bandwidth charge issues CMCC is facing. Last but not least, the solution to the problem is analysed.

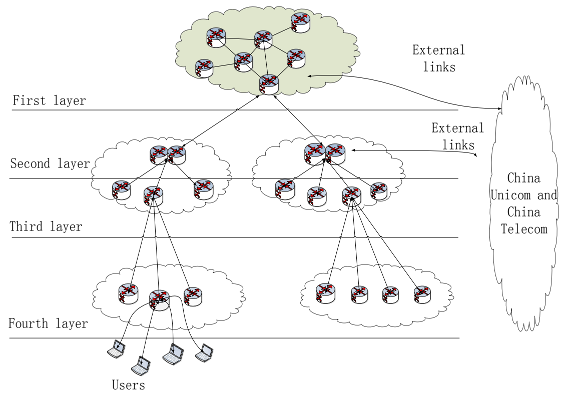


Fig. 3. Network topology of CMCC

A. Network Topology of CMCC

As shown in Fig. 3, CMCC has seven interconnected core nodes in the nation. These nodes are located in China's seven major cities including Beijing, Wuhan, Shanghai, Shenyang, Xi'an, Chengdu and Guangzhou. They represent China's various regions, and support network in their own region as the core node. This is the first layer of the whole network.

The second layer is a provincial-level network. Usually, there are two major network nodes in this level. One of them is the capital city of the province and another is a city with better economic development. Two cities are backup to each other and connected to the regional core node. For example, Hangzhou and Ningbo are major nodes of Zhejiang Province and connected to the regional core node Shanghai.

The third layer consists of other cities in the province. They have direct links to both two provincial-level major nodes. Nevertheless, there are no direct links between these cities. For example, the third layer for Zhejiang province consists of cities such as Jinhua, Taizhou, Wenzhou, Huzhou and so on.

The fourth layer is Metropolitan Area Networks (MAN). Similar to the structure of the provincial network, each directly links to the city node and there are no direct links between districts. There are many communities in a district, and each community has about two or three thousand users. There are no extra routers along the way from the user to the MAN node, so we define the MAN nodes as the edge of the CMCC network.

Every node in the CMCC network can be abstracted as a router with high cache capacity and the whole topology of CMCC network is a tree structure as mentioned above. As shown in Fig. 4. Additionally, nodes in the first and second layer have external links to other fixed network service providers, which are China Unicom and China Telecom.

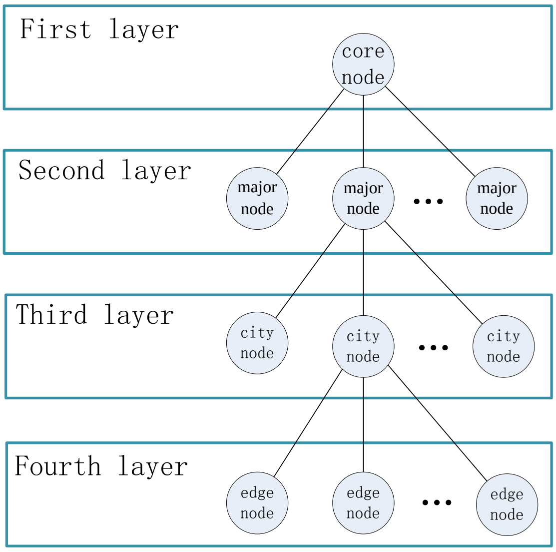


Fig. 4. Tree structure

B. Problem Analysis

CMCC get fixed network license in 2013. As it is the last one to obtain the license of three fixed network service providers in China, the main content providers have been attracted by China Unicom and China Telecom. So contents in CMCC network are pretty poor. And only by the external links from each layer node to China Unicom and China Telecom networks can we get the contents. Specially, the bandwidth of external links is very low compared to the huge demand for contents. Such constraints will greatly reduce the user experience. In addition, bandwidth traffic charges of external links are relatively high. It is also a heavy burden for CMCC. This situation will not be improved in the short term without cooperation or content cache for nodes in CMCC. So a high performance ICN/CCN architecture with appropriate storage decision algorithm is the urgent needs of CMCC.

Reducing traffic of trunk links can avoid congestion and will improve the entire network performance. The reduction of load on bandwidth of external links will cut down the enormous bandwidth traffic charges and alleviate congestion. Decreasing average access hops will give users a better experience. Thus reducing average access hops, traffic of trunk links and load on bandwidth of external links are three main objectives in this CMCC's CCN tree structure. These also meet the demand for energy saving [25] for the network.

C. Solutions

CMCC network is a tree structure and has external links in the first and second layer as mentioned above. Which means contents that are not in the CMCC network can be got from both layers' external links and spread to the whole network with the ICN architecture. And choosing which layer's external links to obtain the contents is a major problem. We assume the cost from the first layer to the second layer is C_S . And the whole cost from the deep nodes in a province to the second layer is C_P . And the number of provinces which are in charge of the first layer core node is n . So the cost for getting contents from the first layer is C_1 in (2), and from the second layer is C_2 in (3).

$$C_1 = n(\mathrm{C_S} + \mathrm{C_P}) \tag{2}$$

$$C_2 = C_P + (\mathrm{n}-1) \times (\mathrm{C_P} + 2\mathrm{C_S}) \tag{3}$$

Let minus , we can get (4):

$$C_2 - C_1 = (\mathrm{n}-2) \times C_S \tag{4}$$

Usually there are more than two provinces in a region of China. So getting contents from first layer's external links is a more appropriate way.

Just as the above discussion, the first layer core node can be seen as the content provider in CCN. And the bottom edge nodes are content consumers. Obviously, reducing average access hops, traffic of trunk links and load on bandwidth of external links is similar to the objectives of in-network caching. Caching high popular contents closer to consumers can reduce the access hops, and this will also alleviate the traffic of trunk links. Avoiding redundant content within the Internet can let more contents be cached in CMCC network and thus reduce the number of requests from the external network. And the idea of our algorithm exactly is to reduce content redundancy within network and make popular contents near clients. This idea is suitable for the need of CMCC.

V. Experiments and Discussion

In this section, the experimental results of the collaborative caching algorithm are given and discussed.

A. Simulation Settings

The proposed algorithm is evaluated by ndnSIM simulator [26], an NS-3 module that implements Named Data Networking (NDN) communication model. We change the ndnSIM process pipeline to implement the popularity estimate and the proposed algorithm. Our experimental machine is a DELL OptiPlex 9020 with intel i7-4790 CPU, and 4 GB RAM.

As mentioned in Section I, delivering the content, reducing the traffic and relieving server load are three main objectives for caching in network. We use the average hop count h as the metric to illustrate the effect of the algorithms on delivering the content and reducing the traffic. Specifically, the average hop count can also be treated as hop count proportion α in (5) to more accurately represent the efficiency of the algorithm. And we use the server hit proportion β in (6) to reflect the effect on relieving server load.

$$\alpha(\mathrm{t}) = \frac{\sum_{ct=1}^{CT} h_{ct}(\mathrm{t})}{\sum_{ct=1}^{CT} H_{ct}(\mathrm{t})} \tag{5}$$

where $H_{ct}(\mathrm{t})$ is the whole path count of Interest for content ct from a client to a server from $t-1$ to t, and $h_{ct}(\mathrm{t})$ is the path count from a client to cache hit router along the way. Obviously, $H_{ct}(\mathrm{t}) = h_{ct}(\mathrm{t})$ if there is no matching cache along the way.

$$\beta(\mathrm{t}) = \frac{\sum_{ct=1}^{CT} r_{ct}(\mathrm{t})}{\sum_{ct=1}^{CT} R_{ct}(\mathrm{t})} \tag{6}$$

where $R_{ct}(\mathrm{t})$ is the whole number of Interests for content ct during $[t-1,t]$, and $r_{ct}(\mathrm{t})$ is the number of Interests which hit servers.

K-ary trees, which represent strict regular structure, will be used in our simulation. Besides k-ary trees, we use Barabasi–Albert (BA) model [27] to generate random scale-free network to evaluate our algorithm. Different network topologies can estimate each algorithm deeply in different ways.

Content requests are generated following Zipf-Mandelbrot distribution with q equals 0.7 and s equals 0.7 by default. The total number of different contents that will be requested by clients is 1,000. The Interests of each client node are generated following Poisson distribution with λ equals 200. Cache size of each router in the network is 100, and total simulation time is 300s. We use the Least Recently Used (LRU) [17] as the cache replacement algorithm, which is widely used as it is efficient and rather simple to be implemented. And the forwarding strategy is flooding as default.

Our popularity-based collaborative caching algorithm (PCC) is compared with other caching algorithms: Leaving Copies Everywhere (LCE) [16], Leave Copy Down (LCD) [18], Leave Copy at edge (LCEdge) [14]. LCE is the default cache decision strategy of CCN. When the content returns, all nodes along the way cache it. LCD only caches the content in the next hop of the hit node. And this makes contents with high popularity come near consumers eventually. LCEdge only caches the contents at the edge of the network.

B. *Experiments with Trees*

A k-ary tree has two main parameters, named as K and D. K denotes the number of child nodes of each father node, and D denotes the depth of the tree. The root node of the tree is used as the content server and the bottom leaf nodes of the tree are client nodes.

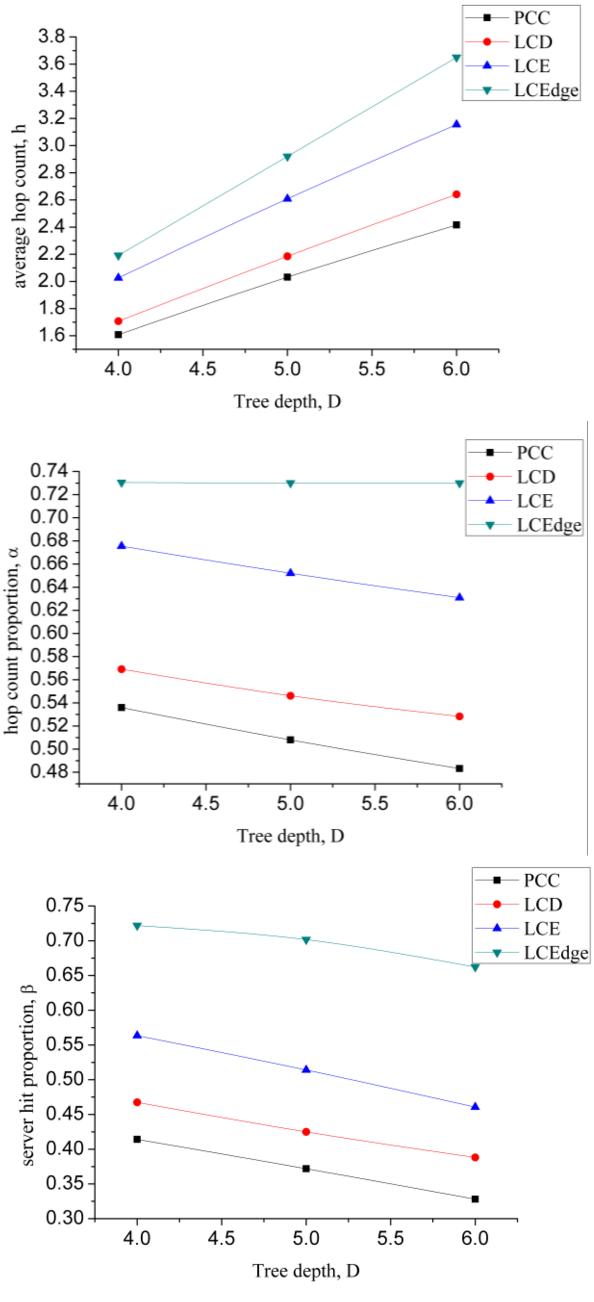


Fig. 5. Effect of tree depth

We first show the effect of binary tree depth on performances in Fig. 5. For average hop count, our algorithm has the smallest hop count. And rising rate of PCC is smaller than comparison algorithms with increasing depth. For hop count proportion, PCC also performs better. LCEdge remains stable as there are only edge nodes cache contents. And it is not affected by depth changes. For server hit proportion, simulation result reflects the advantage of PCC for reducing server load. In summary, our algorithm PCC gets the better performance in average hop count, hop count proportion and server hit proportion than comparison algorithms. And the performance distance between our algorithm PCC and comparison algorithms gets larger with the growth of the depth. As the number of nodes along the path from client to server increases when the depth increases, and less redundant content and popular contents near client with our algorithm, PCC can better use in-network caches.

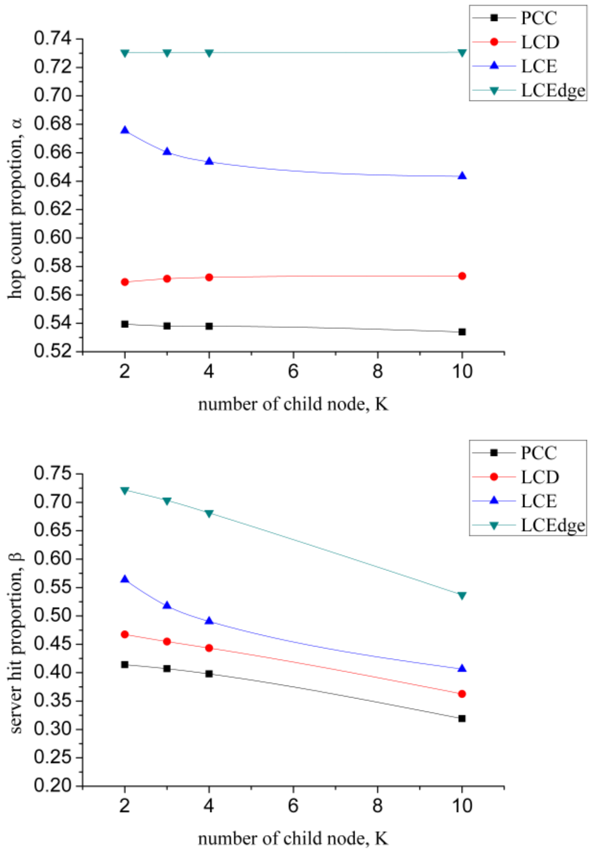


Fig. 6. Effect of K

As shown in Fig. 6, we fix depth to four and explore the effect of K on performances. We increase K from two to four to see the effect of K on performances. We also choose K equals ten as each node in the first to the third layer of CMCC network has about ten child nodes. The k-ary tree with $K=10$ and $D=4$ is similar to the topology of CMCC. Average hop count and hop count proportion are same here as the depth is fixed to four in this simulation. So we just show hop count proportion and server hit proportion in Fig. 6. Our algorithm remains to have the best performances for both metrics as shown. And for our algorithm, hop count proportion keeps stable and server hit proportion steadily declines with the increase of K.

As mentioned above, we use the local popularity, which is measured by each node independently without additional interactive information between nodes, to compare in our algorithm. And we also introduce the global popularity. We can simply assume that the global popularity of the content is fixed. We do some changes on ndnSIM so that global popularity of each content can be identified by routers with content's index. Specific instantaneous performance analysis between global popularity and local popularity is shown in Fig. 7 with $D=4$ and $K=10$. As the depth is still fixed to four, we also just show hop count proportion and server hit proportion as above. As simulations in this paper begin with no cache in routers (cold start) and no Interest counts, algorithms need time to reach the stable state. As shown in Fig. 7, our PCC algorithms based on local popularity and global popularity both perform much better than other algorithms for all metrics at a stable state. As the global popularity is fixed with content and can easily be got by routers in simulation, PCC-global reach stable state faster than PCC-local. In addition, PCC-local performs close to PCC-global for hop count proportion and the performances for server hit proportion of both are almost the same. The global popularity can better reflect the real network traffic, but it is pretty hard to get in the real world. And PCC with local popularity can still perform similar to PCC with global popularity. We can see that local popularity which can be easily estimated by each router is enough to be used in our algorithm.

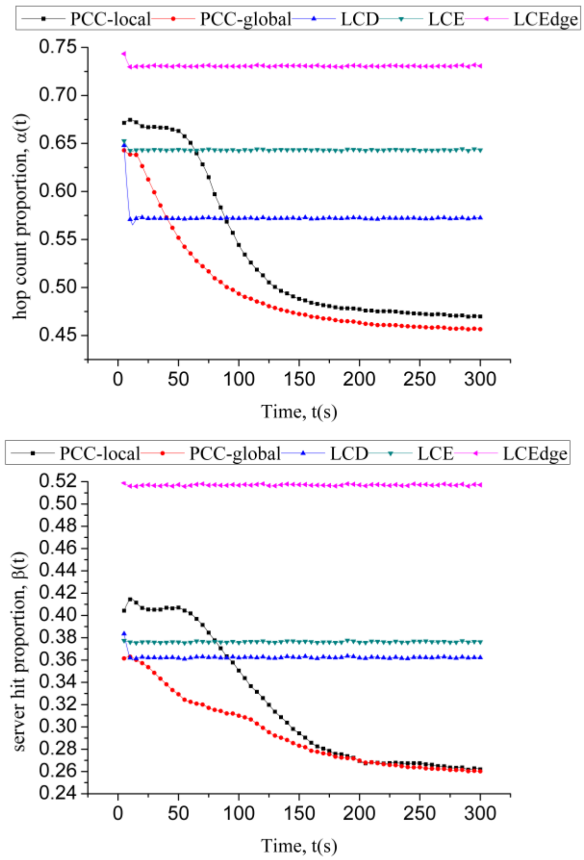


Fig. 7. Instantaneous performance analysis

C. Experiments with BA Graph

Scale-free topologies are concerned in this paper to reflect some real-world topologies that are not regular. And we use the open source software BRITE [28] to generate BA graph. Parameter settings are : $N=100$,

$m = 2$, growth type is incremental. The node with the largest input and output degree is selected as the server. And 20 nodes are randomly selected as clients.

The experiment topology is shown in Fig. 8. The performances of all contrasting algorithms are shown in Table II. Our algorithm also gains better performance than the comparison algorithms. Relative to LCD, which has the best performance among the comparison algorithms, our PCC algorithm is about 8% lower at average hop count and has about 13% performance improvement at server hit proportion.

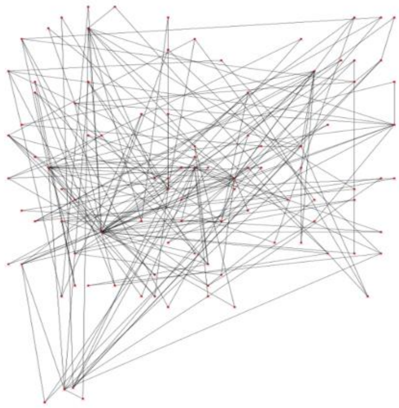


Fig. 8. BA graph for experiment

TABLE II: METRICS FOR BA GRAPH EXPERIMENT

Metric	PCC	LCD	LCE	LCEdge
average hop count	1.01501	1.09362	1.24251	1.26061
server hit propotion	0.47864	0.54473	0.59831	0.70986

D. Discussion

From the above simulation results, we can get the following conclusions. Our algorithm always outperforms the comparison algorithms in all metrics for both tree and scale-free topologies. Thus our algorithm is effective in delivering the content, reducing the traffic and relieving server load. The proposed algorithm shows great scalability with the tree topologies. The algorithm can get better performances with the increasing tree depth. And the algorithm can also reduce server load effectively with increasing K for tree topologies. The performance of our collaborative caching algorithm gets no worse with variation in simulations. The algorithm can keep stable for changing situations. The proposed algorithm based on real-time analysis local popularity has the similar performance with preinstall global popularity. And this conclusion proves the practicability of our algorithm considering local popularity estimate is much more convenient.

Comprehensively considering the effect of D and K, our algorithm gets effective, scalable and stable performances for CMCC network topology, especially proved by the experiment with $D = 4$ and $K = 10$. As discussed above, the proposed algorithm will perform excellently with more or less child nodes for each layer and possible deeper path in real CMCC network.

LCEdge only caches the content at the edge of the network, and intermediate caches are not used. As the edge cache is limited, LCEdge gets the worst performance in the experiments. LCE caches the content along the path and intermediate caches are used relative to LCEdge. LCE performs better than LCEdge as Interests may get response from intermediate caches. LCD also uses all in-network nodes and redundant content is reduced. Additionally, popular contents are near clients with LCD. These make LCD better than LCE and LCEdge in experiments. But LCD still leaves redundant cache along the path. As shown in Section III, our algorithm can avoid redundant content within network and cache more popular contents near the consumers. These are why our algorithm is effective than baseline methods.

While extra time and space are needed for popularity compare and tag process in our algorithm. The proposed algorithm accurately locates contents in network based on the popularity compared to other algorithms. And the redundant content is reduced with the collaborative mechanism. These contribute to the improvement of the performance.

The contents are assumed individual in this paper. What is the performance of our algorithm with the contents are not individual and related ? It is interesting to explore this in future research.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, a novel collaborative caching algorithm is proposed to better utilize in-network caching of CCN. The algorithm is lightweight and implicit collaborative with real-time analysis local popularity and little change to the pipeline of CCN. We also analyze and simulate that local popularity which can be easily estimated by each router is enough to be used in our algorithm compared to the complex global popularity. The proposed algorithm attains the expected performance with the real-time analysis local popularity. The algorithm is simulated with many factors that may have impacts on the performance, and our algorithm is effective, scalable and stable for various topologies and features. And the proposed algorithm is also suitable for solving the lack of contents and high external bandwidth charge problems of CMCC.

We have proved the good performance of our collaborative caching algorithm based on the content popularity. In the future, we will explore the scalability of our algorithm for the situation that the contents are related.

ACKNOWLEDGMENT

This work was supported by the Special Fund for Strategic Pilot Technology Chinese Academy of Sciences (No. XDA06010302). The authors would like to thank the anonymous reviewers for their valuable comments to improve the quality of the paper.

REFERENCES

[1] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, *et al.*, "A survey of information-centric networking research," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 2, pp. 1024-1049, Second Quarter 2014.

[2] S. Li, Y. Shao, S. Ma, N. Xue, S. Li, *et al*., "Flexible Traffic Engineering (F-TE): When OpenFlow meets multi-protocol IP-forwarding," *IEEE Communications Letters*, vol. 18, no. 10, pp. 1699-1702, Oct. 2014.

[3] Z. Zhu, S. Li, and X. Chen, "Design QoS-Aware multi-path provisioning strategies for efficient cloud-assisted SVC video streaming to heterogeneous clients," *IEEE Transactions on Multimedia*, vol. 15, no. 4, pp. 758-768, Jun. 2013.

[4] N. Xue, X. Chen, S. Li, L. Gong, D. Hu, *et al*., "Demonstration of OpenFlow-Controlled network orchestration for adaptive SVC video manycast," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1617-1629, Sept. 2015.

[5] M. Gritter and D. R. Cheriton, "An architecture for content routing support in the internet," in *USITS*, 2001, pp. 1-4-4.

[6] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. H. Kim, *et al*., "A data-oriented (and beyond) network architecture," in *Proc. Conference on Computer Communications*, 2007, pp. 181-192.

[7] G. García, A. B. Ben, F. J. Ramón, A. Maeso, I. Psaras, *et al*., "COMET: Content mediator architecture for content-aware networks," in *Future Network & Mobile Summit (FutureNetw)*, 2011, pp. 1-8.

[8] J. L. Drury and D. J. Mooney, "Hydrogels for tissue engineering: Scaffold design variables and applications," *Biomaterials*, vol. 24, no. 24, pp. 4337-4351, Nov. 2003.

[9] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, *et al*., "Networking named content," in *Proc. 5th International Conference on Emerging Networking Experiments and Technologies*, 2009, pp. 1-12.

[10] M. Yamamoto, "A survey of caching networks in content oriented networks," *IEICE Transactions on Communications*, vol. E99B, no. 5, pp. 961-973, May 2016.

[11] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, *et al*., "Named data networking," *Computer Communication Review*, vol. 44, no. 3, pp. 66-73, July 2014.

[12] S. Podlipnig and L. Böszörmenyi, "A survey of web cache replacement strategies," *ACM Computing Surveys (CSUR)*, vol. 35, no. 4, pp. 374-398, Dec. 2003.

[13] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," in *Proc. 1st International Conference on Information-Centric Networking*, 2014, pp. 127-136.

[14] A. Dabirmoghaddam, M. M. Barijough, and J. Garcia-Luna-Aceves, "Understanding optimal caching and opportunistic caching at the edge of information-centric networks," in *Proc. 1st International Conference on Information-Centric Networking*, 2014, pp. 47-56.

[15] G. Tyson, S. Kaune, S. Miles, Y. E. Khatib, A. Mauthe, *et al*., "A trace-driven analysis of caching in content-centric networks," in *Proc. 21st International Conference on Computer Communications and Networks*, 2012, pp. 1-7.

[16] N. Laoutaris, S. Syntila, and I. Stavrakakis, "Meta algorithms for hierarchical web caches," in *Proc. 23rd IEEE International Performance, Computing, and Communications Conference, Conference Proceedings*, 2004, pp. 445-452.

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

[18] V. Martina, M. Garetto, and E. Leonardi, "A unified approach to the performance analysis of caching systems," in *Proc. 33rd IEEE Conference on Computer Communications*, 2014, pp. 2040-2048.

[19] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, *et al*., "Wave: Popularity-based and collaborative in-network caching for content-oriented networks," in *Proc. Computer Communications Workshops (INFOCOM WKSHPS)*, 2012, pp. 316-321.

[20] A. Xu, X. Tan, and Y. Tian, "Design and evaluation of a utility-based caching mechanism for information-centric networks," in *Proc. of IEEE ICC'15*, London, 2015, pp. 5535-5540.

[21] Z. Ming, M. Xu, and D. Wang, "Age-based cooperative caching in information-centric networking," in *Proc. 23rd International Conference on Computer Communication and Networks*, 2014, pp. 1-8.

[22] A. Ioannou and S. Weber, "Towards on-path caching alternatives in information-centric networks," in *Proc. IEEE 39th Conference on Local Computer Networks*, 2014, pp. 362-365.

[23] J. Li, H. Wu, B. Liu, J. Lu, Y. Wang, *et al*., "Popularity-driven coordinated caching in named data networking," in *Proc. 8th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, 2012, pp. 15-26.

[24] I. Solis, M. Mosko, G. Scott, and A. Walendowski, "CCNx 1.0 tutorial: Theory and practice," in *Proc. 2nd International Conference on Information-Centric Networking*, 2015, p. 1.

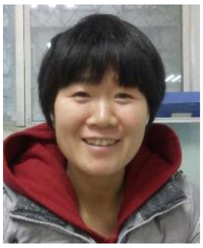
[25] Z. Zhu, P. Lu, J. J. P. C. Rodrigues, and Y. Wen, "Energy-Efficient wideband cable access networks in future smart cities," *IEEE Communications Magazine*, vol. 51, no. 6, pp. 94-100, Jun. 2013.

[26] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," *University of California, Los Angeles*, Tech. Rep, 2012.

[27] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509-512, 1999.

[28] A. Medina, A. Lakhina, and J. Byers, "BRITE: An approach to universal topology generation," in *Proc. 9th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2001, pp. 346-353.

Xiaodong Zhu received his Bachelor's degree from Nanjing University of Aeronautics and Astronautics, Nanjing, China in 2013. He is currently a Ph.D candidate with National Network New Media Engineering Research Center, Institute of Acoustics, Chinese Academy of Sciences, Beijing, China. His research fields include Information Centric Networking, Software Defined Network, etc.



Weining Qi received her Master's degree from The PLA Information Engineering University, Zhengzhou, China, in 2005, and Ph.D. degree in signal and information systems from Institute of Acoustics, Chinese Academy of Sciences, Beijing, China, in 2008. She is currently a post-doctorate researcher in National Network New Media Engineering Research Center, Chinese Academy of Sciences, Beijing, China. Her current research interests are smart systems, routing and information centric networking.

Jinlin Wang graduated from Mathematics Department of University of Science and Technology of China with his bachelor degree in 1986. His research fields include digital signal processing, future networks, network media, etc. He is the Director of Network and New Media Technology Research Center.

Yiqiang Sheng received his Master's degree from Nankai University, Tianjin, China, in 2003 and his Doctoral degree from Tokyo Institute of Technology, Tokyo, Japan, in 2014. He worked at Sanyo Electric Co., Ltd. in Japan from 2003 to 2008 and at Nokia Corporation in Tokyo, Japan, from 2008 to 2010. He did research at Osaka University, Osaka, Japan, from 2010 to 2011 and at Tokyo Institute of Technology, Tokyo, Japan, from 2011 to 2014. He is currently with National Network New Media Engineering Research Center, Chinese Academy of Sciences, Beijing, China, as academic researcher and associate professor. His current research interests are in smart systems, optimization algorithms, machine learning, big data and network theory with applications.