Adaptive Caching Strategy Based on Big Data Learning in ICN

Ling Cai[1], Xingwei Wang[2], Keqin Li[3], Hui Cheng[4], Jiannong Cao[5]

[1] School of Control Engineering, Northeastern University at Qinhuangdao, China
[2] College of Software, Northeastern University, China
[3] Department of Computer Science, State University of New York, USA
[4] Department of Computer Science, Liverpool John Moores University, UK
[5] Department of Computing, The Hong Kong Polytechnic University, China
cailing@126.com, wangxw@mail.neu.edu.cn, lik@newpaltz.edu, H.Cheng@ljmu.ac.uk, csjcao@comp.polyu.edu.hk

Abstract

In-network caching, a typical feature of information centric networking (ICN) architecture, has played an important role on the network performance. Existing caching management strategies mainly focus on minimizing the redundancy content by exploiting either node data or content data respectively, which may not lead to effectively improve the caching performance, as there is no consideration on supplementary action of these two types of data. In this paper, the correlation between node data and content data brought by the big data are analyzed and mined to determine whether the selected content are cached in a few suitable nodes, and a Big data driven Adaptive In-network Caching management strategy (BAIC) is proposed. Driven by the current state of node and content, a novel multidimensional state attribution data model including network, node and content data is proposed. Based on the data model, the mapping relationship between the status data and the matching relationship value is further analyzed and mined. And then utilizing this mapping relationship function, the matching algorithm to predict the matching relationship between the node and the content in the next time period is proposed. The simulation experiments demonstrate that the proposed BAIC has significantly improved the network performance.

Keywords: Information centric networking, Caching, Big data learning

1 Introduction

The initial invention of the Internet was to fulfill the users' primary communication and resource sharing needs. However, according to Cisco's report, the most popular function of the Internet is currently to access content. The traditional location-centric mode appears to be unsuited to deal with the change. Then a new Internet architecture, Information Centric Networking (ICN), has been proposed, such as Named Data Networking (NDN) /Content-Centric Networking (CCN) [1], Data-Oriented Network Architecture (DONA) [2], Network of Information (NetInf) [3], and Service Oriented Future Internet Architecture (SOFIA) [4], Content Mediator architecture for content-aware nETworks (COMET) [5].

Obviously, one of the essential characteristic of ICN is in-network caching. Although caching theories and techniques have been extensively studied, these works usually focused on specific applications with the lack of unique identification of content, such as P2P, CDN and Web [6]. However, ICN makes routing and caching decisions on unique content identification and makes the identification network-aware without authenticated by a specific host. These features make ICN is application-agnostic. Consequently, the previous caching theories and techniques are not suitable, new mechanisms and strategies are urgently required.

As an infrastructure service, ICN has to cache massive amounts of content. However, different from traditional server nodes on the network edge such as CDN, the cache space of the routing node is limited [7]. To fully utilize the cache space, some new strategies have been developed. These strategies can be, in general, divided into two categories. The first category is based on node data to select important cache nodes. Cache nodes are chosen according to the topology of the network and the quality of routing nodes. The second category is based on content data to select important content, i.e., whether to cache or not depends on the content popularity.

Although both categories of strategies do help improve the cache performance, they also have their own limitations. The former one cannot detect the content of different popularity on different demands for cache space, while the latter one is unable to evaluate the network performance and allocate the cache space fairly. Because of lack of effective exploiting the supplementary action of these two data, neither of the

*Corresponding Author: Xingwei Wang; E-mail: wangxw@mail.neu.edu.cn
DOI: 10.3966/160792642018111906005

strategies can be easily used to achieve the optimal usage of the cache space. To solve the above problem, the correlation between node data and content data should be taken into consideration.

In recent years, the rapid network expansion and the fast development of the Internet applications have presented a huge challenge to the mutual awareness between node data and content data. The explosive data growth, however, has created golden opportunities for developing an awareness-based caching strategy. Previous researches indicate that the data correlation created by the big data can help allocate resources in a more effective way and in the most rewarding direction [8-9]. So, in this paper, the status data of node and content is taken as a resource. And the mapping relationship between the different states and the matching relationship value is investigated. Then a secure semi supervised support vector machine (S4VM) model is introduced to mine and predict the matching relationship between the node and the content.

In this paper, an adaptive caching strategy is proposed based on big data. The big data from the node and the content is transformed as a driving force for self-adaption. The major contributions of this paper are as follows.

(1) A novel multidimensional state attribution data mode of the nodes and the content driven by big data is proposed to identify and standardize the real-time status of the network, the nodes, and the content.

(2) A new concept named as "viscosity" is proposed. It is used to describe the matching relationship between nodes and content. It can help determine whether a piece of content is suitable to be cached in a certain node.

(3) In the process of matching, S4VM model based on the combination of labeled and unlabeled data is introduced. The model satisfies the precision and effectiveness requirement.

(4) Extensive simulation experiments have been conducted to compare the proposed caching strategy with other four popular strategies under different scenarios. The validity and performance of the proposed caching strategy are examined and evaluated respectively.

2 Related Work

In the early proposal of ICN, CEE (cache everything everywhere) strategy [10] was introduced. Because the content is cached in every node they go through, lots of redundant content are produced and thus many cache space are wasted. In order to improve the utilization of the cache space, a strategy called LCD (leave copy down) [11] was proposed. In LCD, content is moved downwards from its hit node to the one below. In another similar strategy, i.e., the CLS (caching location and searching) strategy [12], the content is moved to the downstream node by a request or moved to the upstream node by the cache eviction, and then one content at most has a chance to be cached on the path between a server and a leaf router.

Some other work has mainly focused on the importance degree of the node. In the betweenness-based strategy [13], content is only cached at downstream nodes which have the biggest betweenness. A new measurement method is put forward in [14] to figure out the importance degree of all the network nodes. Similarly, the method in [15] is based on the node's importance to the community. It measures the node's importance to the community and makes content cached in those nodes where the users can easily access.

To further meet flexibility in nodes selection, some other strategies based on probabilistic models are proposed. A simple strategy in [16], named RAC (random autonomous caching), caches the content in the node with a constant probability p. A probabilistic caching algorithm named ProbCache is proposed in [17-18]. The caching probability is computed by the distance between the node and its source node and the cache capacity. A weighted probability based caching strategy is suggested in [19]. The caching probability is inversely proportional to the distance between the requester and the content. An opportunistic caching strategy proposed in [20], the caching probability of the content is determined by the distance between the node and its source node, as well as its access frequency. A heuristic probability-based caching strategy in [21], called MBP (Max-Benefit Probability-based Caching), the caching probability is proportional to the content popularity and the content placement benefit.

In all the aforementioned strategies, node data are regarded as a primary consideration and the node location is regarded as a main factor. However, the content distribution is not uniform or optimized without considering content data impacting on the caching performance.

Among the caching strategies based on content data, some of the studies focus on the content popularity. An-aged caching strategy is investigated in [22], in order to reduce network delay the popular content is cached in the network edge nodes. A WAVE strategy is proposed in [23]. It studies the correlation among different content, and the number of content cached in one node can be adjusted according to content popularity. Some work further studies the content popularity algorithm. The local popularity considering the total amount of request at that node are weighed and integrated into the overall popularity in [24]. The overall popularity is taken as a criterion for the node to decide whether to cache the content. The strategy in [25] called MPC (Most-Popular Content) computes the content popularity by the number of requests for the content. The strategy in [26], the utilities of the contents which are tracked by network nodes are used

to make the caching decisions. The strategy in [27], called StreamCache, is also a popularity-based policy. The strategy in [28], called PCC (Popularity-based Cache Consistency), guarantees the freshness of cached contents in ICN routers. The utility-driven caching strategy in [29], the utility function of each content is corresponding content hit probability. Because these caching strategies based on content data only discuss how to select the popular content and do not further discuss how to cache the popular content in the reasonable node, large number of nodes may cache the same popular content, the cost of the redundancy content will be high, so that when the popular peak passes, large amounts of the cached popular content will be erased at the same time, causing the unreasonable content oscillation.

So far, a few research works have investigated the influence of the mutual awareness between node data and content data on the cache performance. Both the content popularity and the cache space are considered in [30], the popular content is cached only in some important nodes according to the user's demand and the trend of content popularity. The CC-CCN (cache capacity-aware CCN) strategy in [31] is based on the knowledge of cache space. After analyzing the cache space of each node, the system can select the candidate cache nodes, and distribute the content fairly to each candidate node according to the popularity of the content. The MAGIC (MAx-Gain In-network Caching) strategy in [32] and the PPCCR (Popularity Prediction-based Cooperative Cache Replacement) in [33]are based on the product of the popularity degree and the distance between the node and the source node. In [34], the CRCache (CRoss-layer Caching) algorithm utilizes a cross-layer design to cache content in a number of selected nodes based on the correlation of content popularity and network topology. These researches provide a foundation for the investigation on adaptive caching, but they are incapable of presenting a complete picture of the correlation between these two types of data. In this paper, by analyzing the big data from these two data, the proposed BAIC strategy will determine an optimized caching deployment.

3 System Framework

Based on the control plane and data plane separation mechanism in Software defined networking (SDN) [35], the system framework of the proposed BAIC strategy is shown in Figure 1. The control plane is the decision center. At the control plane, status data component is in charge of acquiring and counting the volume and variety of status data. These data include the network characteristics, node status and content attributes. The knowledge of the natural matching relationship between the nodes and the content and the mapping relationship between the status data and matching relationship value are derived from these data. In the matching relationship analysis component, the matching relationship value is labeled according to the matching degree, if the matching degree is greater than the threshold, the matching relationship value is labeled as 1, and it means that the node is suitable for caching the current content; otherwise, the value is labeled as-1, the node isn't suitable for caching the content. The basic standard of judging the matching relationship is provided in this component. The matching domain knowledge contains the knowledge of all status data when the value is labeled as 1 or -1 respectively. In the mapping relationship component, the status data impacting on the matching relationship value is analyzed and mined. The major function of this component is predicting in the next period what the matching relationship value is according to the status data and then deciding whether to cache the content. The data plane is used to cache content.

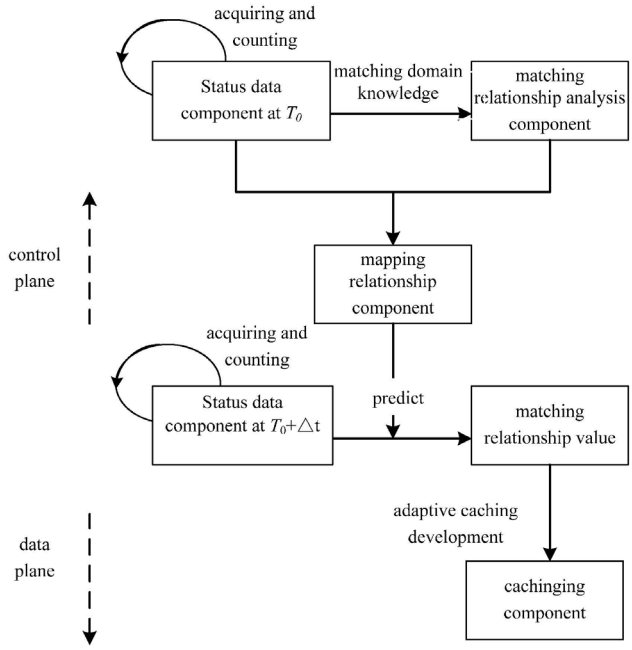


Figure 1. The BAIC system framework

4 Caching Strategy Based on Multidimensional Data Learning

4.1 Extraction of Multidimensional State Attribution Data

The ever-increasing network scale and the wide application of the present Internet lead to a lot of work on the performance measurement of the node and content. The measurement process produces big data, and the analysis of such big data is becoming a huge task. In order to reduce the analysis work on data and obtain the complete and high precision of the node and content description, some reasonable data sample objects need to be selected. Intuitively, the nodes which play more important roles should cache more content. Consequently, we need to define the degree of

node importance. It has been shown that defining the degree of node importance only based on the graph-related centrality properties, such as betweenness, closeness, cannot achieve a good gain on network performance [36]. And nodes have different working conditions at different times. So, the degree of node importance has to consider both the graph-related centrality properties and its own dynamic characteristics. From the perspective of data analysis, in this paper, data are analyzed and mined from following three dimensions, i.e., network dimension, node dimension, and content dimension. In the network dimension, some global statistics of the network including the node weight and the connect degree are analyzed. This dimension is used to decide the degree of node importance based on the graph-related centrality properties. In the node dimension, the cache ratio and cache replacement ratio are calculated for each node respectively. This dimension is used to select the available cache nodes based on the dynamic caching characteristics. In the content dimension, the local popularity and request viscosity of the content are defined. This dimension is used to select the contents that need to be cached. Through the persistent awareness of the above three dimensional data, the comprehensive and high precision data analysis about ICN can be achieved.

4.1.1 Network Dimension

In the network dimension, we analyze the effect on traffic and the range of direct force that the node applies to the adjacent nodes.

The node weight is defined as NW.

$$NW = \frac{CSH}{USH} \tag{1}$$

Here, CSH indicates the number of hops from the cache node to the server node in the path, and USH indicates the number of hops from the user node to the server node along the shortest path.

The reasons for defining the variable CSH are as follows. An example is shown in Figure 2, suppose the content requested by user 1 is responded and cached at node a2, and the content requested by user 2 is responded and cached at node b3, then $CSH(a2) = 2$ and $CSH(b3) = 3$. After that, assume that user 3 requests the same content as previously requested by user 1, and user 4 requests the same content as previously requested by user 2. Because an important goal of the caching in ICN is to reduce the traffic flow, selecting node a2 can only eliminate the request to node a1, but selecting node b3 can reduce the traffic flow of two nodes b1, and b2. Therefore, to reduce the whole network traffic, the weight of node b3 should be bigger than the weight of node a2 in terms of being selected as the cache node.

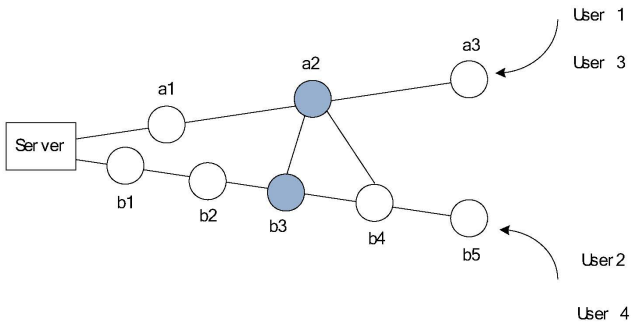


Figure 2. Illustration of the network dimension parameters

For USH, the number of hops requested by user 1 is 3, and the number of hops requested by user 2 is 5. Suppose the cache nodes are a2 and b3. From the perspective of traffic flow, the weight of node b3 should be bigger than the weight of node a2 in terms of being selected as the cache node. However, from the user's perspective, the requests from user 3 and 4 should go through 1 and 2 hops respectively to reach the cache node. Because in ICN, the content is expected to be closer to the user, if we consider the hops only, the weight of selecting node a2 is bigger than the weight of selecting node b3. Therefore, in order to balance the relationship between the traffic flow and the number of hops, the node weight as NW is defined in this paper. The bigger the candidate node's weight is, the more suitable the node is.

The connection degree of a cache node is defined as CD, which is the number of its adjacent links. Considering the topology in Figure 2, the connection degree of a2 is 4 and the connection degree of b3 is 3. It can be used to estimate the range of direct force that the node applies to the surrounding nodes in the network. The bigger the connection degree of a node, the stronger its force range is. As the content is cached at node a2, it can provide the caching service directly to the four neighboring nodes, while the caching service is only available to three neighboring nodes if the cache node is $b3$.

4.1.2 Node Dimension

Node dimension is used to describe the load condition of a node during different time periods, through the cache ratio and the cache replacement ratio respectively.

The node cache ratio is defined as CR.

$$CR = \frac{\sum_{i=0}^{n} CCS_i}{CCS(v)}. \tag{2}$$

Here, CCS represents the size of cached content, n is the number of cached content within unit time, and $CCS(v)$ is the size of cache space of node v. The node

cache ratio can effectively describe the load level of the node whose cache space is not fully utilized.

The node cache replacement ratio is defined as RR.

$$RR = \frac{\sum_{i=0}^{n'} RCS_i}{CSS(v)}. \quad (3)$$

Here, RCS represents the size of the replaced content, n' is the number of replaced content within unit time. Assume that the network is in a stable state and some nodes have their cache spaces occupied. Then the cache replacement ratio can effectively describe the node load and the cache condition, reflecting the timeliness of different content in the node [37]. The combination of node cache ratio and node cache replacement ratio constitutes a set of the node condition.

4.1.3 Content Dimension

For any content, its popularity will experience a dynamic changing process, i.e., the rise, the peak, and the final attenuation. The popularity of content is also influenced by its location and its request ratio, thus even for the same content its popularity on different nodes may not be the same. In the content dimension, both temporal correlation and spatial correlation of the popularity are discussed herein. In the perspective of temporal correlation, the popularity is describing the dynamic changing trend of the number of content requests. In the perspective of spatial correlation, the viscosity is used to define the matching degree between the content of different popularity and the node at different position. Because ICN contains two kinds of packets, namely, interest packet and data packet, the request viscosity of the interest packet is the expectations of the correlation between the content and the nodes.

The local popularity of the content is defined as LP_{vi}.

$$LP_{vi} = \frac{IRN_{vi}}{\sum_{i=1}^{n} RN_{vi}}. \quad (4)$$

Here, IRN_{vi} is the number of interest packet i requests within unit time according to the PIT table [38], n' is the number of interest packet within unit time. $\sum_{i=1}^{n} IRN_{vi}$ is the total number of requests in this node within unit time.

The network popularity of the content is defined as NP_i.

$$NP_i = \sum_{v=1}^{m} LP_{vi}. \quad (5)$$

Here, m is the total number of nodes. Since LP_{vi} computes the local popularity, the dynamic changes of content popularity in the network can be obtained by the statistical analysis of all the nodes. The popularity of the content is governed by Zipf distribution, i.e., the access requests to the top 20% popular content account for 80% of total network requests. The top 20% popular content by NP_i is determined accordingly. During the peak time of the content, LP_{vi} on many nodes is large. If the content happens to be cached in these nodes, many of the same content are in the network, the redundancy cost will be high. Intuitively, the popular content should be cached in the best-matched nodes, different content should be cached in the different nodes. Consequently, a matching relationship between the content and the node is needed to be defined.

The request viscosity of the content is defined as RV_{vi}.

$$RV_{vi} = LP_{vi} \times \log \frac{m}{m(i)}. \quad (6)$$

Here, $m(i)$ is the number of nodes sending interest packet i. $\log \frac{m}{m(i)}$ is a global factor of all nodes and describes the relative importance of interest packet i to all nodes. It only relates to all nodes, regardless of any specific node. When more nodes send the same interest packet, the value becomes smaller, implying a lower correlation between the request and nodes. The request viscosity is proportional to the popularity of the content on the node, and is inversely proportional to the number of requesting nodes in the whole network. The bigger the viscosity value, the higher the correlation between content i and node v is, and the bigger the degree of matching expectations.

4.2 Definition of Matching Relationship Value

The matching relationship value checks whether the content i matches the node v.

The caching response ratio is calculated as:

$$CRN_{vi} = \frac{crn_{vi}}{\sum_{i=1}^{n} crn_{vi}}, \quad (7)$$

where crn_{vi} is response times to cached content i in the current node v within unit time according the CS table [38].

The cache viscosity is defined as CV_{vi}.

$$CV_{vi} = CRN_{vi} \times \log \frac{m}{m'(i)}. \quad (8)$$

Here, $m'(i)$ is the number of nodes caching content

i. $\log\frac{m}{m'(i)}$ is the relative importance of cached content i to all nodes. The matching degree between the cached content and the node is obtained by calculating the cache viscosity. The bigger the cache viscosity value, the greater the matching degree between the node and the content, and then the node is more suitable for the content.

TH is the matching degree threshold value, if the cache viscosity $CV_{vi} \geq TH$, this means that content i has a good matching degree on node v, meaning that content i is suitable to be cached in node v, then the matching relationship value is labeled as 1, otherwise it is labeled as -1.

4.3 Design of the Training Set and the Test set

For each node, the attribute vector is used to describe the state attribute, and defined as:

$$a_{vi} = (NW_{vi}, CD_{vi}, CR_{vi}, RR_{vi}, LP_{vi}, RV_{vi}) = (a_{vi}^1, a_{vi}^2, a_{vi}^3, a_{vi}^4, a_{vi}^5, a_{vi}^6)$$

The 6 element listed from left to right are node weight, connection degree, cache ratio, replacement ratio, local popularity, and request viscosity. $A_{vi}^t = (a_{11}, a_{12} \cdots a_{vi})$ is the labeled data set. A_{vi} is the set of A_{vi}^t over a period of time T_0, where $t \in T_0$. To simplify the representation of the data set, let $A_{vi}^t = X_1 = (x_1, x_2 \cdots x_1)$, where $x_1 = a_{11}$. The category set is $Y_1 = (y_1, y_2 \cdots y_1)$, in which $y_j \in \{-1, 1\}$. X_1 and Y_1 constitute the training set. $X_u = (x_{1+1}, x_{1+2} \cdots x_{1+u})$ is unlabeled data set over a period of time $T_0 + \Delta t$, the category set $Y_u = (Y_{1+1}, Y_{1+2} \cdots x_{1+u})$ is the unknown relationship value, and they are the test data set.

4.4 Cache Resource Management System

In this section, we present a resource management system architecture for controlling the routing processes as well as the cache resources in Figure 3.

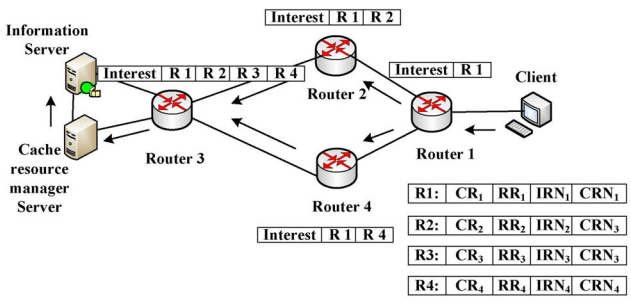


Figure 3. Resource management system architecture

The parameters on the node dimension, like CR and RR, can be calculated by analyzing the cache status in every node. Also, the original parameters on the content dimension, like IRN_v, and crn_v can be obtained by analyzing the Interest package. Then these parameters will be forwarded to the Cache Resource Manager Server as additional content of the Interest packet. Each Cache Resource Manager Server has knowledge of the network topology. NW and CD, which we need on network dimension can be calculated by analyzing the topology of the whole network in the Server. By the multidimensional data and the matching relationship value over a period of time T_0 that have been obtained, the Cache Resource Manager Server will learn and dig the matching relationship, then the matching results over a period of time $T_0 + \Delta t$ will be learned and forwarded to the Information Server. The Data packet attached this matching results will be forwarded along the source path. Then each node will cache the corresponding contents based on the matching relationship.

4.5 Semi-supervised Support Vector Machine S4VM Algorithm

This paper aims to predict the value of y_u in the subsequent time period. To this end, we formulate the question of whether to cache the current content as a Binary classification problem.

The traditional classification method has the weakness of overfitting or performing poorly during the learning process of small samples. Support vector machine (SVM) is a traditional classification method based on VC (Vapnik-Chervonenkis) dimension theory and structural risk minimization principle. However, it is only suitable for the study of supervised learning, and it cannot effectively use the unlabeled data [39].

In ICN, it's usually easy to acquire caching performance data of each node, but it is often very expensive to get all the labeled information about these performance data. If all the performance data is not labeled, the workload can be reduced, but the accuracy of the algorithm cannot be guaranteed because of the lack of priority knowledge of the labeled samples. Considering the above two factors, a secure semi supervised support vector machine (S4VM) [40] model based on the combination of labeled and unlabeled data is introduced.

S4VM model focuses on exploiting multiple candidate low-density separators, the reason is that there usually exist more than one large-margin low-density separators for a few labeled data and abundant unlabeled data, it's hard to decide which one is the best separator based on the limited labeled data, a wrong selection may cause a bad matching degree on the content and the node and result in the provider's economic losses and a degenerated performance.

4.6 Adaptive Caching Strategy

The basic idea herein is to analyze and mine the

historical multidimensional state attribution data. Then the matching relationship between nodes and content is figured out via the mapping function between the state attributes and the relationship values. In detail, the proposed strategy consists of two phases.

Phase A: Gauss method [41] is used to normalize the historical data, so that the different dimensions of historical data are mapped to the same value space. The purpose of this preprocessing step is to eliminate the influence of different data ranges on the mining algorithm. Specific steps are as follows:

First, the normalized value of each attribute data a_{vi}^{p} is defined as

$$a_{vi}^{rp} = 0.5 + \frac{a_{vi}^{p} - \tilde{a}_{vi}^{p}}{2 * a\sigma_{vi}^{p}}, \tag{9}$$

where $\tilde{a}_{vi}^{p}$ is the historic average of the attribute data a_{vi}^{p}, and σ_{vi}^{p} is the standard deviation of the historical data a_{vi}^{p}.

Then, if the data a_{vi}^{rp} is still outside the interval [0,1], the following formula is applied:

$$a_{vi}^{rp} = \begin{cases} 0, a_{vi}^{rp} < 0 \\ 1, a_{vi}^{rp} > 0 \end{cases}. \tag{10}$$

Phase B: The mapping algorithm is used to construct the relationship function between the multidimensional state attribute data and the matching relationship values. The mapping algorithm is summarized in Algorithm 1. At first, the algorithm judges whether node v is suitable for caching content i (line 4-5): if the cache viscosity CV_{vi} is greater than the threshold TH, it means that content i has a good matching degree on node v, and the relationship values is labeled as 1; otherwise, the relationship values is labeled as -1. Then, based on the normalized processing, the S4VM algorithm is used to mine and learn the mapping function between the state data and the relationship values (line 9-12). The mapping function f is constructed according to the training set. Assume that the space of the state data is R^{e}, the space of the relationship values is R^{g}, then the mapping function is computed as follows:

$$f: R^{e} \to R^{g} = x \to w'\theta(x) + b \tag{11}$$

where $w'\theta(x) + b$ is a set of mapping functions composed of g sub functions $\{f_1(x), ...f_g(x)\}$, and each sub function represents the mapping relationship between the state attribute values and the relationship value. The matching relationship values among the content and nodes can be predicted and labeled by inputting the state attribute values in the next period $T_0 + \Delta t$ (line 15) to the mapping function f. The calculation is conducted with the following formula:

$$Y_u = w'\theta(X_u) + b \tag{12}$$

If the labeled relationship value is labeled as 1, and then the content is cached in the node, otherwise, the content isn't cached in the node (line 16-19).

Algorithm 1. Mapping algorithm

Input: the state attribute value $\{x_j\}_{j=1}^{l}$ and $\{x_u\}_{l+1}^{l+u}$, the threshold value TH, the relationship value $\{y_j\}_{j=1}^{l}$, the cache viscosity $\{CV_j\}_{j=1}^{l}$, the space of the state attribute value R^{e}, the space of the relationship values R^{g}.

Output: the relationship value $\{y_u\}_{l+1}^{l+u}$ at the moment $T_0 + \Delta t$.

1: Set N be the number of all content; M be the number of all nodes; $x_j = a_{vi}$, $K_j = k_{vi}$, $x_u = a_{(v+1)(i+1)}$, $i \in N$ and $v \in M$; Set $\{y_j\}_{j=1}^{l} = 0$; Set T be the time;
2: if $T = T_0$ then
3: for $j = 1$ to l do
4: if $K_j \geq TH$, then y_j =1;
5: else y_j =0;
6: end if
7: end for
8: end if
9: for $j = 1$ to l do
10: $y_j = w'\theta(x_j) + b$;
11 : $f: R^{e} \to R^{g} = x_j \to y_j$;
12: end for
13: if $T = T_0 + \Delta t$ then
14: for $j = l + 1$ to $l + u$ do
15: $Y_u = w'\theta(X_u) + b$;
16: solving by the S4VM algorithm, if y_u =1, then
17: cache the content i in the node v;
18: else
19: do not cache the content i in the node v;
20: end if
21: end for
22: end if
23: return $\{y_u\}_{l+1}^{l+u}$;

5 Simulation and Analysis

5.1 Performance Metrics and Experimental Setup

The primary objectives of caching strategies are to: (1) reduce network operating cost, the amount of traffic,

the number of cached content and the number of requests to the server; (2) improve the quality of user experience so that users can quickly acquire various content from the nearly nodes.

To quantify the above caching objectives, performance metrics are introduced for the network and the user, respectively. The network operation cost is analyzed from three aspects, i.e., the network, the routing nodes and the servers. In addition, the average link utilization, cache ratio and server load ratio are defined, respectively. To analyze the quality of the user experience, the cache hit ratio, delay, hop reduction ratio, and content diversity ratio are defined, respectively. In the following, we list the performance metrics defined in this paper and their brief explanation.

Average link utilization: average value of all link utilization in the network within unit time.

Cache ratio: the ratio of the number of cached content to the total number of request content.

Server load ratio: the ratio of the number of requests received by all servers to the number of requests sent by all users within unit time.

Cache hit ratio: the ratio of the number of hit cache content to the total number of cached content.

Hop reduction ratio: the ratio of the reduced number of hops from the client to the first node where a successful cache hit occurs to the number of hops from the client to the server.

Delay: the time from sending interest packet to receiving data packet.

Content diversity ratio: the ratio of the number of different content cached in the node to the total number of different content produced by the servers in the network.

In this paper, real domain topology AS-1755 [42] which is publicly available through Rocketfuel is resorted to. The requests for different content are generated based on Zipf-distribution with α=0.7. The average user request ratio is 100 packets per second. The total number of different content in the network is 71,000. This paper has analyzed only about 14,500 content which are in the top 20%. One content needs one cache space. The total cache space of the network is from 0.25G to 1.5G. The initial amount of cached content in each node is zero. To evaluate our proposed adaptive caching strategy BAIC, the performance evaluation is conducted with ndnSIM in which all the nodes are cache enabled. Four caching strategies, namely, CEE [9], LCD [10], Prob [16] and BETW [11], are compared with the proposed strategy. The nodes use the least frequently used (LFU) replacement algorithm [43].

During simulations, we repeat experiments 10 times and each time has 4000,000 request events. When the network enters into a steady state phase after 1000,000 request events, we collect multidimensional state data and cache viscosity per 100 request event. After subsequent 1500,000 events, we labeled these cache viscosities as matching relationship values, and then these status data and relationship values constitute the training set. We implement BAIC strategy to mine the mapping relationship with the training set, then input the status data of the last 1500,000 request events as test data, the output is the matching relationship between the content and the node.

5.2 Performance Evaluation

5.2.1 Impact on the Network Operating Cost

We firstly investigate the network operating cost gain of five cache strategies under different cache capacity. Figure 4 shows the average link utilization ratio with varying cache capacity. First and foremost, we can see that the link utilization ratio of every caching strategy decreases with the increase of the cache capacity. For example, when the cache capacity is 0.25G, the link utilization ratio of BAIC is nearly 23%, when the cache capacity is 1.5G, the link utilization ratio of BAIC can be reduced to 14%. The reason is that with the increase of the cache capacity, the amount of cached content is also increased. The user can obtain the required content at the intermediate cache nodes, thus reducing the traffic flow from the cache nodes to the server, so the average link utilization is reduced as well. Further analysis shows that the performance of BAIC strategy is better than the other three. For a 1.5G cache capacity, BAIC reduces the link utilization ratio by about 30% compared to LCD and about 40% compared to CEE, Prob0.5 and BETW. Although a large amount of content is cached in the nodes under the other four strategies, they don't consider the matching relationship between the content and the nodes. If the content does not meet the user needs, the users still need to get the required content from the server.

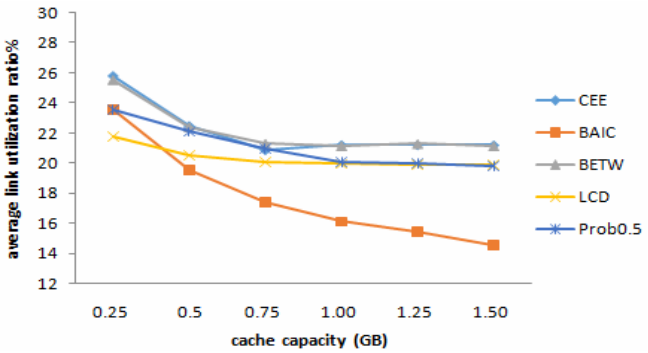


Figure 4. The average link utilization ratio under different cache capacities

Figure 5 shows the cache ratio under the five different caching strategies within unit time. From the figure, we can see that with the increase of the cache capacity, the performance of all strategies is significantly improved, except LCD. For example, when the cache capacity is 0.25G, more than 20% of the content should be cached in the network with BAIC.

However, when the cache capacity is 1.5G, the cache ratio can be reduced to 8%. However, in LCD strategy, it is required to cache not only a large amount of new content but also cache the content from the upstream nodes to the downstream nodes, so the change trend is different from the other three strategies at the sampling time. Moreover, BAIC strategy significantly outperforms the other four strategies. For a 1.5G cache capacity, in terms of the cache ratio, BAIC reduced the cache ratio by about 66% compared to BET and Prob0.5 and about 78% compared to CEE and LCD.

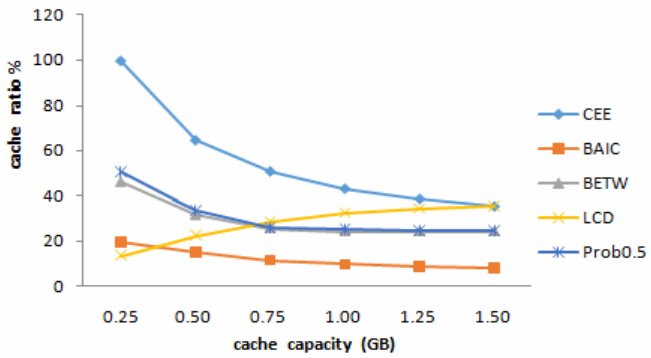


Figure 5. The cache ratio under different cache capacities

Figure 6 shows the server load ratio under the five caching strategies within unit time. BAIC strategy decreases the server load ratio by about 14.1%-39.3% compared to CEE, about 16.8%-40.5% compared to BETW, 19.2%-64% compared to Prob0.5 and about 28.6%-55.7% compared to LCD. As mentioned previously, in BAIC strategy, it caches the popular content in the matching nodes. As a result, the request has a higher chance of hitting the desired content in the routing nodes than in other strategies.

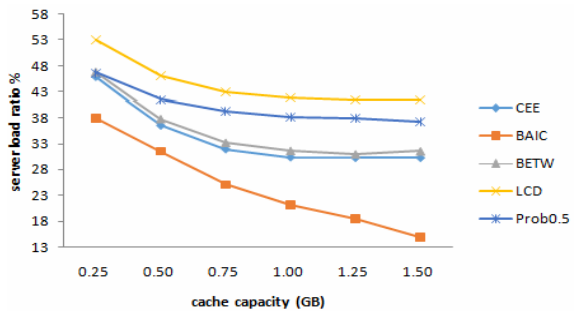


Figure 6. The server load ratio under different cache capacities

5.2.2 Impact on the Quality of User Experience

We investigate the quality of user experience gain of five cache strategies under different cache capacity. In Figure 7, BAIC strategy improves the cache hit ratio by nearly 40% compared to CEE, Prob0.5, BETW and also remarkably exceeds LCD when the cache capacity is 1.5G. The reason is analyzed as follows. In CEE strategy, there are a large number of redundant content which does not match the nodes, so the cache hit ratio is relatively low. Meanwhile, due to limited cache space, content is constantly being replaced which further affects the cache hit ratio. In BETW strategy, content is cached only at those more "important" nodes along the paths, therefore a relatively limited number of content can be cached in these "important" nodes. Similar to BETW, in Prob strategy, only some content can be cached. In LCD, the number of cached content is less than the number in any other strategy, and thus the cache hit ratio is the lowest.

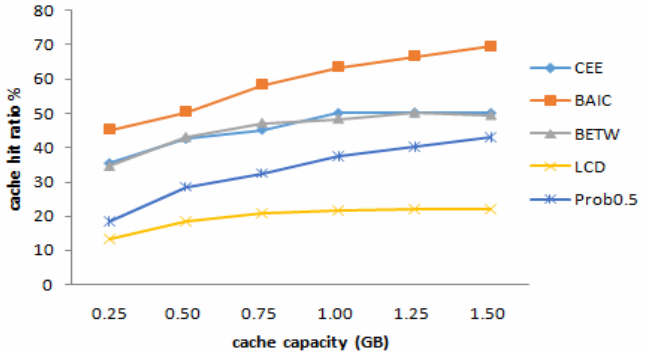


Figure 7. The cache hit ratio under different cache capacities

Figure 8 shows the hop reduction ratio under the five different caching strategies. For each of them, the hop reduction ratio increases with the increase of cache capacity. Moreover, BAIC strategy significantly outperforms the other four strategies. For a 1.5G cache capacity, BAIC improves the hop reduction ratio nearly by 30% compared to the other three. And with the increase of the cache capacity, the performance gap between BAIC and the other strategies is widened.

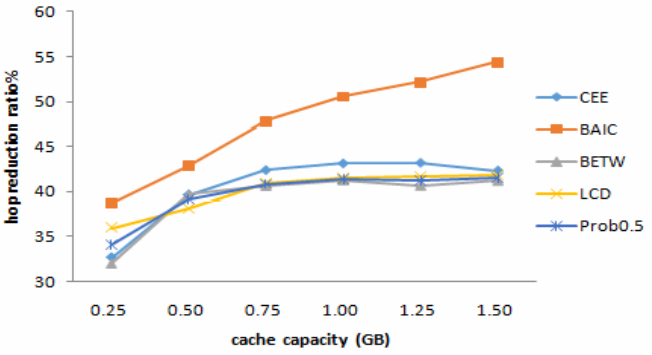


Figure 8. The hop reduction ratio under different cache capacities

Figure 9 shows the access delay under the five different caching strategies. When the user is able to obtain the required content directly from the cache node, the hop count is reduced and the access delay is reduced as well.

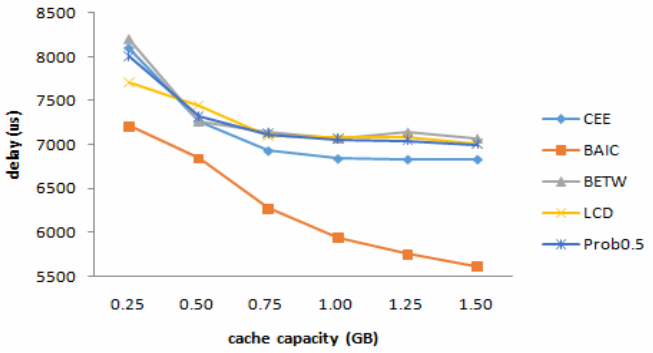
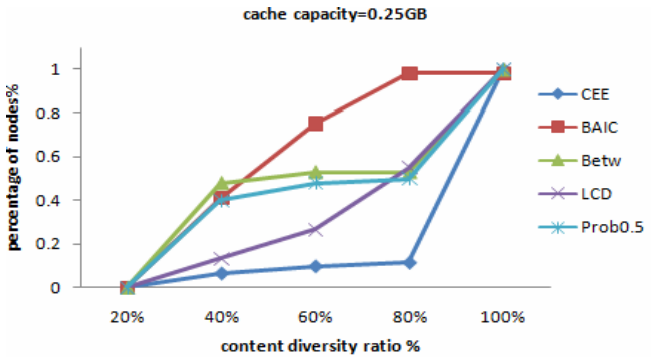
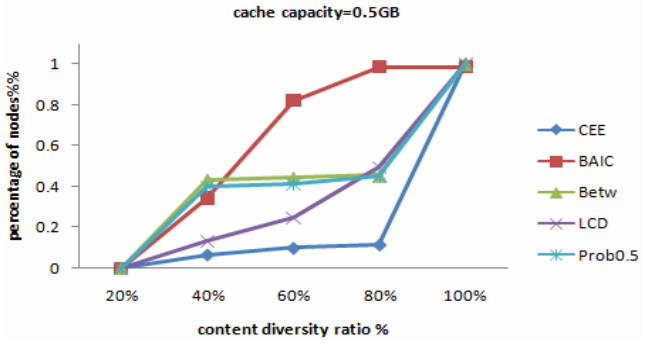


Figure 9. The delay under different cache capacities

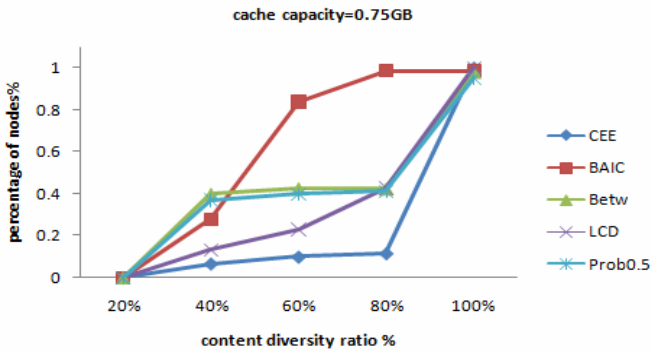
Figure10 shows the content diversity ratio of the five strategies under different cache capacities. In order to analyze the diversity of the cache content, the horizontalaxis represents content diversity ratio, and the vertical axis represents the percentage of the number of nodes. We can see that to the same strategy the distribution of the content diversity is roughly similar under different cache capacities. In Figure 10 (a), (b) and (c), to BAIC strategy, about 40% of the network nodes have the content diversity ratio less than 40%, about 80% of the network nodes have the percentage of total content less than 60%, and nearly all the network nodes have the content diversity ratio less than 80%. While in Figure 10(d), (e) and (f), the content diversity ratio of BAIC is reduced. The reason is that the matching degree or the cache viscosity is inversely proportional to the number of nodes caching content *i* in the whole network, with the increase of the cache capacity, the number of nodes caching content *i* is increased, then in the training process, the



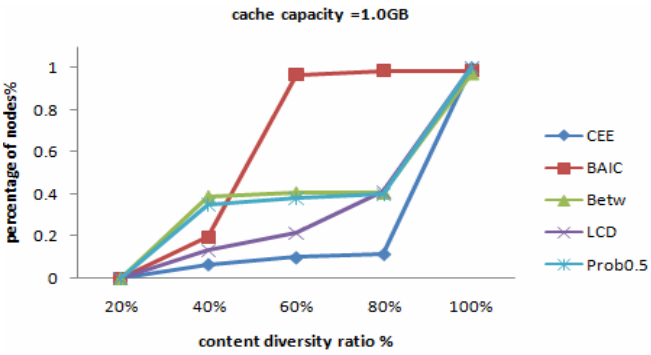
(a) cache capacity=0.25GB



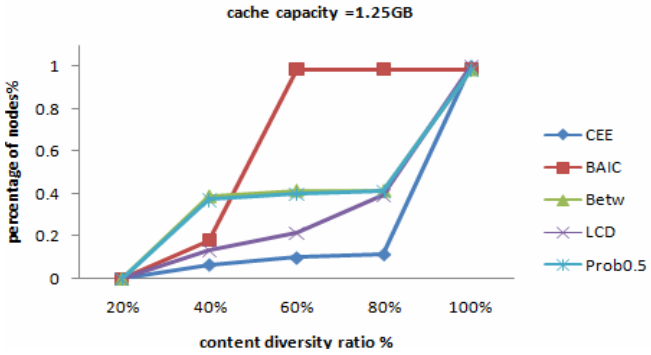
(b) cache capacity=0. 5GB



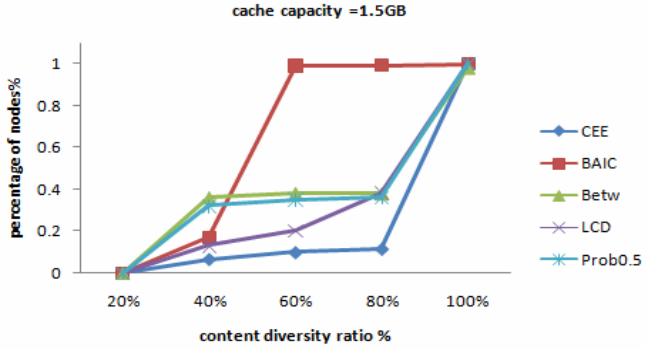
(c) cache capacity=0.75GB



(d) cache capacity=1GB



(e) cache capacity=1.25GB



(f) cache capacity=1. 5GB

Figure 10. The content diversity ratio under different cache capacities

number of matching degree greater than the threshold is reduced, it means that only the content which matching degree with the nodes are strong are suitable to be cached in the current nodes. In the test process, the number of content cached in the node is reduced under this strong matching constraint.

5.2.3 Impact of Zipf Parameter (α)

Figure 11 shows the cache hit ratio under BAIC strategy of difference cache capacity for difference values of Zipf parameter α. With the increasing of Zipf parameter, the content with higher popularity will be cached on more nodes, then the cache hit ratio increases.

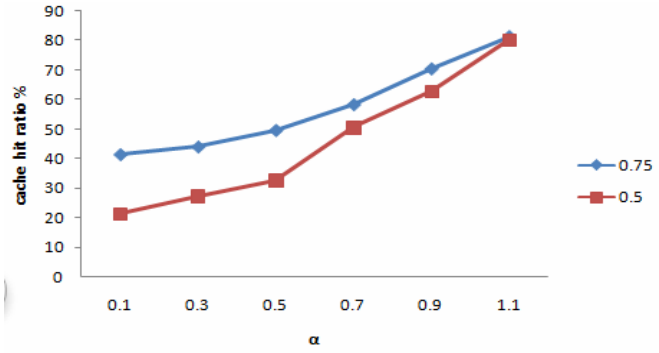


Figure 11. cache hit ratio vs Zipf parameter

5.2.4 Impact of Replacement Algorithm

Figure 12 shows the cache hit ratio under BAIC strategy of difference cache capacity for difference replacement algorithm. From the figure, we can see that with the increase of the cache capacity, the performance of two replacement algorithms is significantly improved. The reason why LFU algorithm always performs worse than LRU is because when a content is received again after a deletion, it can be deleted easily since it has a low reference frequency compared with other content. Therefore once a content is deleted from the cache on the node, it's difficult to be cached again, and the request has a hardly chance of hitting the desired content in this node.

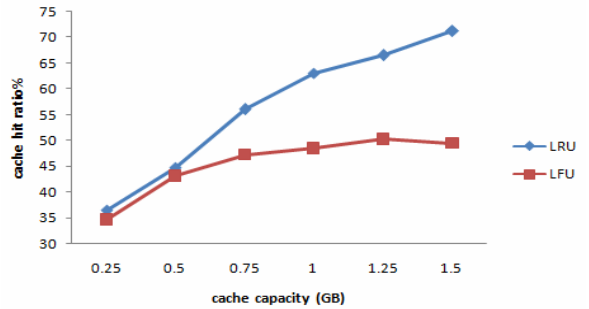


Figure 12. cache hit ratio for LRU and LFU replacement algorithm

6 Conclusion

In this paper, the cache management strategy in ICN architecture is investigated. In order to cache the content in the suitable nodes, BAIC strategy is proposed to predict the matching relationship between the nodes and content. The big data learning process is built upon analyzing and mining the historical big data of multidimensional state attribution, i.e., the network, node and content. Then, the prediction is conducted by the mapping function between the state attributes data and the caching relationship values with S4VM. Experimental results show that this caching strategy has significantly reduced the network operating cost and improved the user experience quality. We conclude that by fully analyzing and mining the correlation of node data and content data, cache performance could be significantly enhanced.

Acknowledgements

This work is supported in part by Science and Technology Research project of University of Hebei Province under Grant No.QN2014327, the National Science Foundation for Distinguished Young Scholars of China under Grant No. 61225012 and No. 71325002.

References

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, R. L. Braynard, Networking Named Content, *Proc. of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT'09)*, Rome, Italy, 2009, pp. 1-12.

[2] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, I. Stoica, A Data-oriented (and Beyond) Network Architecture, *Proc. of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'07)*, Kyoto, Japan, 2007, pp. 181-192.

[3] C. Dannewitz, J. Golic, B. Ohlman, B. Ahlgren, Secure Naming for a Network of Information, *Proc. INFOCOM IEEE Conference on Computer Communications Workshops*, San Diego, CA, 2010, pp. 1-6.

[4] G. Xie, Y. Sun, Y. Zhang, Z. Li, H. Zheng, X. Zheng, Demo Abstract: Service-Oriented Future Internet Architecture (SOFIA), *Poster of IEEE Conference on Computer Communications (INFOCOM)*, Shanghai, China, 2011, pp. 1-2.

[5] W. K. Chai, N. Wang, L. Psaras, G. Pavlou, C. Wang, G. G. de Blas, F. J. Ramon-Salguero, L. Liang, S. Spirou, A. Beben, E. Hadjioannou, CURLING: Content-ubiquitous Resolution and Delivery Infrastructure for Next-generation Services, *IEEE Communications Magazine*, Vol. 49, No. 3, pp. 112-120, March, 2011.

[6] G. Q. Zhang, Y. Li, T. Lin, Caching in Information Centric

Networking: A Survey, *Computer Networks*, Vol. 57, No. 16, pp. 3128-3141, November, 2013.

[7] D. Perino, M. Varvello, A Reality Check for Content Centric Networking, *Proc. of the ACM SIGCOMM Workshop on Information-centric Networking (ICN '11)*, Toronto, ON, Canada, 2011, pp. 44-49.

[8] T. Wolf, J. Griffioen, K. L. Calvert, R. Dutta, G. N. Rouskas, I. Baldine, A. Nagurney, Choice As a Principle in Network Architecture, *ACM SIGCOMM Computer Communication Review*, Vol. 42, No. 4, pp. 105-106, October, 2012.

[9] H. Yin, X. Zhang, T.-Y. Zhan, Y. Zhang, G. Min, D. O. Wu, NetClust: A Framework for Scalable and Pareto-optimal Media Server Placement, *IEEE Transactions on Multimedia*, Vol. 15, No. 8, pp. 2114-2124, December, 2013.

[10] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, J. Wilcox, Information-centric Networking: Seeing the Forest for the Trees, *Proc. of the 10th ACM Workshop on Hot Topics in Networks (HotNets-X)*, Cambridge, MA, 2011, pp. 1-6.

[11] N. Laoutaris, S. Syntila, I. Stavrakakis, Meta Algorithms for Hierarchical Web Caches, *Proc. IEEE International Conference on Performance, Computing, and Communications (IPCCC)*, Phoenix, AZ, 2004, pp. 445-452.

[12] Y. Li, T. Lin, H. Tang, P. Sun, A Chunk Caching Location and Searching Scheme in Content-centric Networking, *Proc. of IEEE International Conference on Communications (ICC)*, Ottawa, ON, Canada, 2012, pp. 2655-2659.

[13] W. K. Chai, D. He, I. Psaras, G. Pavlou, Cache "Less for More" in Information-centric Networks, *Proc. of the 11th International IFIP TC 6 Conference on Networking (IFIP'12) - Volume Part I*, Prague, Czech Republic, 2012, pp. 27-40.

[14] Y. He, Y. Zhu, Y. Ni, J. Shi, N. Zhu, A Cache Strategy in Content-centric Networks Based on Node's Importance, *Information Technology Journal*, Vol. 13, No. 3, pp. 588-592, March, 2014.

[15] J. Cai, S.-Z. Yu, W.-X. Liu, Caching Strategy Based on Node's Importance to Community in Information-centric Networks, *Journal on Communications*, Vol. 36, No. 6, pp. 1-10, June, 2015.

[16] S. Arianfar, P. Nikander, J. Ott, On Content-centric Router Design and Implications, *Proc. of the Re-Architecting the Internet Workshop (ReARCH'10)*, Philadelphia, Pennsylvania, 2010, pp. 51-56.

[17] I. Psaras, W. K Chai, G. Pavlou, Probabilistic in-network Caching for Information-centric Networks, *Proc. of the Second Edition of the ICN Workshop on Information-centric Networking (ICN '12)*, Helsinki, Finland, 2012, pp. 55-60.

[18] I. Psaras, W. K Chai, G. Pavlou, In-Network Cache Management and Resource Allocation for Information-Centric Networks, *IEEE Transactions on Parallel & Distributed Systems*, Vol. 25, No. 11, pp. 2920-2931, November, 2014.

[19] L. Saino, I. Psaras, G. Pavlou, Hash-routing Schemes for Information-centric Networking, *Proc. of the 3rd ACM SIGCOMM Workshop on Information-centric Networking (ICN '13)*, Hong Kong, China, 2013, pp. 27-32.

[20] X. Hu, J. Gong, Opportunistic On-path Caching for Named Data Networking, *IEICE Transactions on Communications*, Vol. E97.B, No. 11, pp. 2360-2367, November, 2014.

[21] H. Wu, J. Li, J. Zhi, MBP: A Max-Benefit Probability-based Caching Strategy in Information-Centric Networking, *Proc. of IEEE International Conference on Communications (ICC)*, London, UK, 2015, pp. 5646-5651.

[22] Z. Ming, M. Xu, D. Wang, Age-based Cooperative Caching in Information-centric Networks, *Proc. of IEEE INFOCOM Workshops*, Orlando, FL, 2012, pp. 268-273.

[23] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, S. Pack, WAVE: Popularity-based and Collaborative In-network Caching for Content-oriented Networks, *Proc. of IEEE INFOCOM Workshops*, Orlando, FL, 2012, pp. 316-321.

[24] H. Wu, J. Li, T. Pan, B. Liu, A Novel Caching Scheme for the Backbone of Named Data Networking, *Proc. of IEEE International Conference on Communications (ICC)*, Budapest, Hungary, 2013, pp. 3634-3638.

[25] C. Bernardini, T. Silverston, O. Festor, MPC: Popularity-based Caching Strategy for Content Centric Networks, *Proc. of IEEE International Conference on Communications (ICC)*, Budapest, Hungary, 2013, pp. 3619-3623.

[26] A. Xu, X. Tan, Y. Tian, Design and Evaluation of a Utility-based Caching Mechanism for Information-centric Networks, *Proc. of IEEE International Conference on Communications (ICC)*, London, UK, 2015, pp. 5535-5540.

[27] W.-J Li, S. M. A. Oteafy, H. S. Hassanein, StreamCache: Popularity-based Caching for Adaptive Streaming over Information-centric Networks, *Proc. of IEEE International Conference on Communications (ICC)*, Kuala Lumpur, Malaysia, 2016, pp. 1-6.

[28] B. Feng, H. Zhou, H. Zhang, J. Jiang, S. Yu, A Popularity-based Cache Consistency Mechanism for Information-Centric Networking, *Proc. of IEEE Global Communications Conference (Goblecom)*, Washington, DC, 2016 pp. 1-6.

[29] M. Dehghan, L. Massoulie, D. Towsley, D. Menasche, Y.Tay, A Utility Optimization Approach to Network Ca-che Design, *Proc. of IEEE INFOCOM 2016- The 35th Annual IEEE International Conference on Computer Communications*, San Francisco, CA, 2016, pp. 1-9.

[30] W.-X. Liu, S.-Z. Yu, X. Hu, P.-Y. Zhu, Selective Caching in Content-Centric Networking, *Chinese Journal of Computers*, Vol. 37, No. 2, pp. 275-288, February, 2014.

[31] D. Kim, S.-W. Lee, Y.-B. Ko, J.-H. Kim, Cache Capacity-aware Content Centric Networking under Flash Crowds, *Journal of Network and Computer Applications*, Vol. 50, pp. 101-113, April, 2015.

[32] J. Ren, W. Qi, C. Westphal, J. Wang, K. Lu, S. Liu, S. Wang, MAGIC: A Distributed MAx-Gain In-network Caching Strategy in Information-Centric Networks, *Proc. of IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Toronto, ON, Canada, 2014, pp. 470-475.

[33] W. Zhao, Y. Qin, D. Gao, C. H. Foh, H.-C. Chao, An Efficient Cache Strategy in Information Centric Networking Vehicle-to-Vehicle Scenario, *IEEE Access*, Vol. 5, pp. 12657-12667, June, 2017.

[34] W. Wang, Y. Sun, Y. Guo, D. Kaafar, J. Jin, J. Li, Z. Li, CRCache: Exploiting the Correlation between Content Popularity and Network Topology Information for ICN Caching, *Proc. of IEEE International Conference on Communications (ICC)*, Sydney, NSW, Australia, 2014, pp. 3191-3196.

[35] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, T. Turletti, A Survey of Software-defined Networking: Past, Present, and Future of Programmable Networks, *IEEE Communications Surveys & Tutorials*, Vol. 16, No. 3, pp. 1617-1634, Third Quarter, 2014.

[36] D. Rossi, D. Rossini, On Sizing CCN Content Stores by Exploring Topological Information, *Proc. of IEEE INFOCOM Workshops*, Orlando, FL, 2012, pp. 280-285.

[37] X.-D. Cui, J. Liu, T. Huang, J.-Y. Chen, Y.-J. Liu, A Novel In-network Caching Scheme Based on Betweenness and Replacement Rate in Content Centric Networking, *Journal of Electronics & Information Technology*, Vol. 36, No. 1, pp. 1-7, January, 2014.

[38] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, L. Zhang, A Case for Stateful Forwarding Plane, *Computer Communications*, Vol. 36, No. 7, pp. 779-791, April, 2013.

[39] Z.-H. Zhou, M. Li, Semi-Supervised Learning by Disagreement, *Knowledge and Information Systems*, Vol. 24, No. 3, pp. 415-439, September, 2010.

[40] Y.-F. Li, Z.-H. Zhou, Towards Making Unlabeled Data Never Hurt, *IEEE Transactions on Pattern Analysis And Machine Intelligence*, Vol. 37, No. 1, pp. 175-188, January, 2015.

[41] M. Ortega, Y. Rui, K. Chakrabarti, S. Mehrotra, T. S. Huang, Supporting Similarity Queries in MARS, *Proc. of the Fifth ACM International Conference on Multimedia*, Seattle, Washington, 1997, pp. 403-413.

[42] N. Spring, R. Mahajan, D. Wetherall, Measuring ISP Topologies with Rocketfuel, *Proc. of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Pittsburgh, Pennsylvania, 2002, pp. 133-145.

[43] D. Lee, J. Choi, J.- H. Kim, S. H. Noh, S. L. Min, Y. Cho, C. S. Kim, On the Existence of a Spectrum of Policies that Subsumes the Least Recently Used (LRU) and Least Frequently Used (LFU) Policies, *ACM SIGMETRICS Performance Evaluation Review*, Vol. 27, No. 1, pp. 134-143, June, 1999.

Biographies

Ling Cai received the BS and MS degrees from Harbin institute of technology, China, in 2002 and 2004, respectively, and PhD degree from the Northeastern University, China. She is currently a lecturer at the Northeastern University at Qinhuangdao. Her research interests include future internet and big data, etc.

Xingwei Wang received the B.S., M.S., and the Ph.D. degrees in computer science from the Northeastern University, Shenyang, China, in 1989, 1992, and 1998, respectively. He is currently a professor at Northeastern University. His research interests include cloud computing and future Internet, etc.

Keqin Li received the B.S. degree from Tsinghua University, China, in 1985, and the Ph.D. degree from the University of Houston, TX, in 1990. He is currently a SUNY distinguished professor in the State University of New York at New Paltz. His research interests include cloud computing, big data, etc.

Hui Cheng received the B.S. and the MS degrees from Northeastern University, China in 2001 and 2004, and the Ph.D. degree from The Hong Kong Polytechnic University, Hong Kong in 2007. He is a Senior Lecturer in Liverpool John Moores University. His research interests include cloud computing, optical networks, etc.

Jiannong Cao received the B.S. degree from Nanjing University, China, in 1982, and the M.S. and Ph.D. degrees from Washington State University, Pullman, in 1986 and 1990, respectively. He is currently a Professor in the Hong Kong Polytechnic University, Hong Kong. His research interests include distributed computing, networking, etc.