Designing Content Placement of CDN for Improving Aggregation Effect of ICN FIBs

Yu Sasaki*, Noriaki Kamiyama*, and Yusheng Ji†
*Faculty of Engineering, Fukuoka University, Fukuoka 814-0180, Japan
Email: tl161252@cis.fukuoka-u.ac.jp, kamiyama@fukuoka-u.ac.jp
†National Institute of Informatics, Tokyo 101-8430, Japan
Email: kei@nii.ac.jp

***Abstract*—Information-centric networking (ICN) has attracted wide attention as a new network architecture which can efficiently deliver digital content and Internet of Things (IoT) data. However, prefix aggregation of FIB (forwarding information base) of ICN routers is difficult because the organization names are independent of location. In this paper, we propose to use the CDN (content delivery network) as a method of allocating originals of content at routers to effectively reduce the size of FIBs of ICN routers. Using the measured data of web-content location, we evaluate the proposed algorithm of allocating web content at routers and show that the proposed method can reduce the size of FIBs by about 45%.**

I. Introduction

Traffic generated by delivering video content including user generated content (UGC), e.g., YouTube, and rich content produced by content providers, e.g., movie and dramas, has dominated a large part of traffic on the Internet. Moreover, Internet of Things (IoT) is becoming a reality in smart homes, smart buildings and smart cities, which produce huge amounts of sensor readings that need to be processed to control actuators. As a new network architecture efficiently delivering video content and sensor data of IoT, information-centric networking (ICN), which caches content at routers and routes packets using content name, has attracted wide attention thanks to various merits [6]. To realize the idea of ICN, various networks, such as TRIAD [8], content-centric networking (CCN) [12], data-oriented network architecture (DONA) [13], and named data networking (NDN) [21], have been proposed [20]. In this paper, we assume NDN as the architecture of ICN.

In NDN, packets of requesting content items are called *Interests*. Similar to IP routers of the Internet, NDN routers transfer Interests by looking up the FIB (forwarding information base). Although the FIB in IP routers is matching table between the prefix of destination IP address, i.e., the network address, and the output port, the FIB in NDN router is matching table between the prefix of content name and the output face[1]. Content name is a concatenation of prefix and content ID, e.g., "fukuoka-u/sample.jpg"where "fukuoka-u" is a prefix, and "sample.jpg" is a content ID. In NDN, originals of content exist at hosts of publishers, and publishers advertise the prefix of content name to adjacent routers [10]. NDN Routers receiving the prefix advertisement configure FIBs so that an Interest sent from the subscriber, i.e., user, is transferred to the host of publisher owing the requested content.

If we make entries for all the possible prefixes in the FIBs, the required memory size of FIBs seriously increases. Therefore, aggregating FIB entries is indispensable to reduce the memory cost of FIBs. Network providers assign a part of their assigned address block to their customer organizations, so the IP addresses have the geographical locality, and the IP addresses are hierarchical structure, i.e., IP addresses in nearby areas have identical bit patterns in the upper digit. On the other hand, although hosts of publishers providing content of the same organization tend to exist in the same area, there is no geographical locality in the name of organizations. Therefore, prefix aggregation in NDN is more difficult compared with that in IP networks. For example, when considering web pages as content, there are about 10^{11} content names, and there are about 10^9 prefixes of content name after aggregating prefixes of the same organization into one prefix [7]. Because prefix aggregation among different organizations is difficult, about 10^9 entries are necessary in each FIB in NDN [7], whereas just about 10^5 entries are necessary in each FIB in IP networks. When using the entry lookup based on the hash tables [19], several million M bytes memory is required in FIB even for several million prefixes, so implementing FIBs using SRAM is difficult.

In the Internet, CDN (content delivery network) in which content items are delivered from cache servers provided in many networks has been widely used to improve the quality of content delivery and reduce the amount of traffic in networks [15]. Using NDN, content items are more likely delivered from nearby routers to users, so the purpose of CDN, i.e., improving user quality and reducing the network traffic, is satisfied by NDN. Cache servers of CDN can also advertise prefixes of cached content to the networks [3], and cache servers of CDN can also become the hosts of publishers. Therefore, in this paper, we propose to improve the aggregation effect of FIB entries of NDN routers by carefully allocating copies of content at cache servers of CDN. The contribution of this manuscript is summarized as follows.

- By browsing many webpages, we measure the location of content of webpages, and we evaluate the aggregation degree of FIB entries based on the measured location of

[1]In NDN, an output port of a router is called output face.

web content.
- We propose an algorithm allocating web content at network nodes so that the required size of FIBs of NDN is effectively reduced.
- For the URLs obtained by browsing many webpages, we apply the proposed algorithm allocation web content at network nodes, and we evaluate the reduction effect of NDN FIBs.

In Section II, we briefly summarize the related works, and we show the results of aggregating FIB entries of NDN based on the measured location of web content in Section III. In Section IV, we show the proposed algorithm of allocating web content at nodes and show the numerical results. Finally, we conclude this manuscript in Section V.

II. Related Works

We can summarize the existing methods of reducing the number of entries required in the FIB of ICN routers into the four categories: (i) partial caching, (ii) route aggregation, (iii) flooding, and (iv) bloom filter. In the first approach, i.e., partial caching, FIB entries are created for just a part of prefixes instead of all the prefixes [1][7]. Afanasyev et al. proposed to use the DNS to resolve the name whose entry did not exist in the FIBs of routers [1]. When the prefix entry of arriving Interest did not exist in the FIB of a router, the router returned the negative acknowledgement to the subscriber, and the subscriber obtained the prefix which existed in the same domain and whose entry existed in router FIBs by the DNS. Moreover, Detti et al. also proposed the look and cache approach, i.e., routers obtain the routing information from the NRS (name routing system) server and cache the obtained routing information at the FIBs of routers [7]. In this approach, although the size of FIBs is reduced, the name look-up procedure is required.

In the second approach, i.e., routing aggregation, FIBs are set so that all Interests traverse the identical router called NAC (name collector) [18]. Because Interests transmitted on the tree topology, in which the NAC is the root node, toward the NAC, the required size of FIBs at routers is reduced. However, the path stretch, i.e., the hop length of Interest transmission, increases. In the third approach, i.e., flooding, Interests are broadcast to all output faces at routers without looking up the FIBs [3][5]. Ascigil et al. proposed to broadcast the Interest at routers, and the adjacent routers caching the prefix returned the router information to the requesting router [3]. Chiocchetti et al. proposed to transfer Interests for non-popular content items, whereas broadcast Interests for popular content items without using FIBs [5]. Because copies of popular content are likely cached at many routers, so Interests will arrive at routers having the requested content with high probability by broadcasting Interests. However, Interests are transferred redundantly, so the network links might be overloaded.

Finally, in the fourth approach, i.e., bloom filter, routers judge whether transfer each arriving Interest or not to each output face using the bloom filter provided at each output face [17][16]. Using the bloom filter, routers can make the decision of Interest transfer by using a small size memory with a limited number of memory accesses. However, we cannot avoid a false positive, i.e., falsely transferring Interests to incorrect output faces, and the network load will increase due to redundant Interest transmission.

III. Measuring Location of Web Objects

Alexa provides the list of the top 500 webpages with the largest request count in each of the 16 categories: adult, arts, business, computers, games, health, home, kids&teens, news, recreation, reference, regional, science, shopping, society, and sports [2]. We selected 8,000 webpages consisting of the most popular 500 webpages in each of the 16 categories as the measurement targets. When accessing webpages, some web browsers, e.g., Google chrome, have a function to export the HTTP archive (HAR) files which include various communication properties, e.g., the host URL from which each object is downloaded, the URL of each object, the size of each object, and the delay caused in obtaining each object, as JavaScript Object Notation (JSON) format [4][9]. We continuously and automatically accessed the 8,000 webpages from a client terminal at the Fukuoka University in November 2017 by using the headless Chrome, i.e., Chrome without using GUI. We successfully obtained the HAR files from 7,604 webpages among the 8,000 webpages accessed[2]. The cumulative number of web objects[3] included in the 7,604 webpages was 679,380, and we extracted URLs of these objects from the HAR files.

Because CDN has been widely used in the Internet, many web objects were delivered from cache servers instead of origin servers. We want to focus on the location of only originals of content, so we exclude URLs which were delivered from cache servers. When using the CDN, DNS query was redirected the DNS server of CDN providers, so DNS record included the character string of *CNAME*. Therefore, for each extracted URL, we obtained the DNS record by using *dig* command, and we obtained 20,921 URLs from 679,380 URLs excluding the URLs whose DNS records include the character string of CNAME. Moreover, we further excluded URLs including the five keywords related to CDN: cdn, edge, akam, cloudfront, and cloudflare. 2,374 URLs include any of these keywords, so we had 18,547 URLs for which web objects were delivered from origin servers.

We obtained the country name, city name, and coordinates of the host from which each of the 18,547 objects was transmitted by using the GeoIP API provided by MaxMind [14]. We classified the 18,547 objects based on the country where the original of each object existed, and Table I shows the number of URLs classified into each of the top 10 countries. We confirmed that originals of 18,547 web objects were provided from 71 countries, and those of about 64% web

[2]For various reasons, HAR files cannot be obtained for 396 webpages. For example, timed out error occurred when obtaining some objects, or IP addresses cannot be resolved for some URLs of objects.

[3]Web object is a unit of data embedded in a webpage, and each webpage consists of about 50 to 100 web objects on average.

objects were provided from USA. In the following evaluation, we used the coordinates and URLs of these 11,908 web objects whose originals existed in USA.

In the numerical evaluation, we used the topology of Internet 2 which is the backbone network in USA aiming for research and education [11]. We used 12 nodes providing the layer 3 service, and Fig. 1 shows the topology among the selected 12 nodes. Let O_m denote the node at which the original of web object m is placed among the 12 nodes of the Internet 2 topology. We set O_m to the node to which the Euclidean distance from the coordinate of host delivering object m was minimum among the 12 nodes. Table II summarizes the number of web objects allocated to each of the 12 nodes among the 11,908 objects, and we confirmed that many web objects were provided at the west coast and the east coast as well as the central location of USA, i.e., Kansas City.

TABLE I
NUMBER OF URLS IN EACH OF TOP 10 COUNTRIES

Country	URL count	Country	URL count
United States	11,908	Czech Republic	237
Germany	477	France	228
Korea	322	Hong Kong	222
United Kingdom	261	Australia	202
Singapore	241	Ireland	128

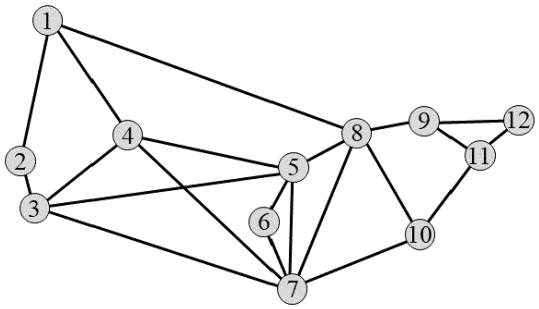


Fig. 1. Layer 3 node topology of Internet2

TABLE II
NUMBER OF WEB OBJECTS WHOSE ORIGINALS ARE ALLOCATED TO EACH NODE

Node	City	Object count
1	Seattle	1,267
2	Sunnyvale	1,997
3	Los Angeles	623
4	Salt Lake City	602
5	Kansas City	3,050
6	Dallas	376
7	Houston	265
8	Chicago	439
9	Cleveland	237
10	Atlanta	322
11	Washington DC	1,611
12	New York	1,119

IV. AGGREGATING FIB ENTRIES USING LONGEST PREFIX MATCHING

In this section, we propose an algorithm to aggregate the FIB entries of NDN routers using the longest prefix matching.

A. Generating Components of URLs

The longest prefix matching is used in IP routers to efficiently reduce the FIB size. FIB entries of IP routers are allowed to take any value, i.e., zero or unity, in the lowest bits of any length, and a packet arriving at an IP router is forwarded to the output port where the number of the highest bits at which the FIB entry and destination IP address of the arriving packet is the maximum. To apply the longest prefix matching to NDN routers, two approaches will be possible: (i) unit of characters of URLs and (ii) unit of components of URLs. We define the components of URL as the character string separated by period in URL. For example, the URL of "www.fukuoka-u.ac.jp" consists of four components, "www", "fukuoka-u", "ac", and "jp". The prefix of NDN is the concatenation of components in the reverse order, i.e., "jp/ac/fukuoka-u/www". Therefore, the first component of URL is the TLD (top level domain), and the second component of URL is the SLD (second level domain).

B. Aggregation Algorithm of FIB Entries

Initially, at each node n, FIB entries were created for all the 11,908 objects excluding objects whose originals were allocated at note n. The proposed aggregation algorithm of FIB entries can be executed at each node independently, so we focus on one node and describe all variables without node index n. Let $P(x)$ denote the prefix of entry x in the FIB, and let $p(x, s)$ denote the first s components of $P(x)$. Moreover, we define $F(x)$ as the output face of FIB entry x which is set by a routing algorithm, e.g., Dijkstra′s minimum cost route. The routing algorithm of Interests is out of the scope of this manuscript, and we assume that $F(x)$ is given. We also define $\boldsymbol{M}$ as the set of FIB entries which have not been checked yet. Now, we describe the algorithm (Algorithm 1) aggregating FIB entries in the forward direction.

Algorithm 1 Aggregating FIB entries in forward direction

```
Initializes s = 1 and M to all FIB entries
while s ≤ C do
    while M ≠ φ do
        Randomly selects one entry x from M
        Calculates Y(x, s, o), set of FIB entry y satisfying
        p(y, s) = p(x, s) with F(y) = o among y ∈ M, for
        each possible o
        Aggregates all entries of Y(x, s, o*) to {p(x, s), o* }
        where number of entries of Y(x, s, o*) is maximum
        among those of Y(x, s, o) of any output face o
        Updates M to M \ {y ∈ Y(x, s, o), ∀x}
    end while
    Increments s
end while
```

Firstly, this algorithm aggregates the FIB entries on the TLDs, and it aggregates the FIB entries on the SLDs in the next step. This procedure is repeated until the aggregation is completed on all the component location, 1, 2, $\cdots$, $C-1$, where C is the maximum number of components in each URL. In the 11,908 web objects, C was six. We also propose to aggregate the FIB entries in the reverse direction, i.e., from the tail C to the top, and we use the same algorithm shown above with setting $s = C - 1$ in step 1, and decrementing s and moving to step 2 when s is greater than unity.

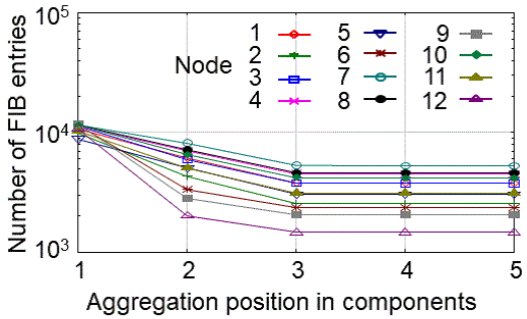
C. Numerical Evaluation

We applied the proposed aggregation algorithm of FIB entries to the measured original location of 11,908 web objects. Let $E^{I}_{o,n}$, $E^{A,f}_{o,n}$, and $E^{A,r}_{o,n}$ denote the number of FIB entries at node n without aggregation, with aggregation in the forward direction, and with aggregation in the reverse direction, respectively. Table III shows $E^{I}_{o,n}$, $E^{A,f}_{o,n}$, and $E^{A,r}_{o,n}$ for each node n as well as $\Delta E^{A}_{o,n}$, the difference between $E^{A,f}_{o,n}$ and $E^{A,r}_{o,n}$, i.e., $E^{A,f}_{o,n} - E^{A,r}_{o,n}$. Even in the measured original location of web objects, the FIB size can be reduced to about 20% to 50% using the proposed aggregation algorithm. The difference of the aggregation effect in both the direction was small. However, the required calculation time of aggregating FIB entries in the forward direction was much smaller than that of aggregating FIB entries in the reverse direction, so the aggregation in the forwarding direction was better than the aggregating in the reverse direction.

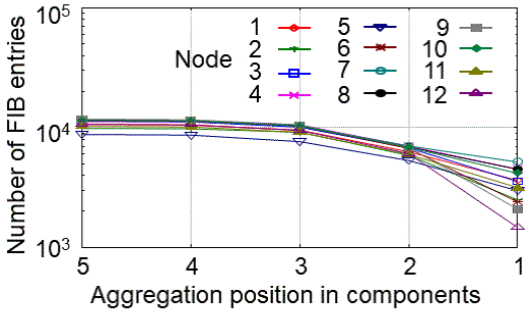
Figure 2(a) plots the number of FIB entries at each aggregation stage, i.e., aggregation point of URL components, at each of the 12 nodes when aggregating the FIB entries in the forward direction. Figure 2(b) also shows the same results when aggregating the FIB entries in the reverse direction. We confirmed that almost all the aggregation effect of FIB entries was obtained when aggregating FIB entries at the first and second components, i.e., TLD and SLD.

TABLE III
NUMBER OF FIB ENTRIES WITHOUT AGGREGATION, $E^{I}_{o,n}$, WITH AGGREGATION IN FORWARD DIRECTION, $E^{A,f}_{o,n}$, AND WITH AGGREGATION IN REVERSE DIRECTION, $E^{A,r}_{o,n}$, AND $\Delta E^{A}_{o,n} \equiv E^{A,f}_{o,n} - E^{A,r}_{o,n}$ BASED ON MEASURED LOCATION OF WEB OBJECTS

Node	$E^{I}_{o,n}$	$E^{A,f}_{o,n}$	$E^{A,r}_{o,n}$	$\Delta E^{A}_{o,n}$
1	10,641	3,792	3,646	146
2	9,911	2,554	2,533	21
3	11,285	3,746	3,583	163
4	11,306	4,528	4,550	-22
5	8,858	3,043	3,011	32
6	11,532	2,360	2,393	-33
7	11,643	5,289	5,207	82
8	11,469	4,567	4,509	58
9	11,671	2,061	2,083	-22
10	11,586	4,196	4,208	-12
11	10,297	3,130	3,139	-9
12	10,789	1,462	1,481	-19



(a) In forward direction



(b) In reverse direction

Fig. 2. Number of FIB entries at each node based on measured location of object originals

V. DESIGNING CONTENT PLACEMENT OF CDN REDUCING FIB SIZES

A. Policy of Designing Content Placement of CDN

As mentioned in the previous section, almost all the aggregation effect of FIB entries was obtained when aggregating FIB entries at the first and second components, i.e., TLD and SLD. Therefore, we focus on designing content placement of CDN cache servers maximizing the aggregation effect of FIB entries on just the TLD and SLD. Figure 3 shows the complementary cumulative distribution (CCD) of web object count with the identical TLD and identical SLD. The number of different TLD and SLD in the 11,908 web objects was 112 and 6,494, respectively. We confirmed that quite few TLDs and SLDs were used in URLs of many web objects, whereas almost all TLDs and SLDs were used in URLs of just a few web objects. Tables IV and V summarize the number of URLs of web objects with each of the top 10 TLDs and the top 10 SLDs, respectively. The deviation of URL count using each TDL was more remarkable compared with that using each SLD, and the TLD of about 70% URLs of the 11,908 web objects was “com”.

Now, let us consider placing each web object at one of the 12 nodes. The identical output face is set to the prefixes destined to the same node in FIB of each node. Therefore, we can expect to efficiently reduce the number of FIB entries by placing web objects with TLDs of high rank at the identical node. As an extreme case, if we place all web objects at just a single node, all FIB entries can be aggregated into just a single entry. However, traffic will be concentrated on the node

with all web objects and its connected links. To alleviate the traffic concentration, we should place web objects at all the 12 nodes as equally as possible. Therefore, we limit the maximum number of web objects placed at one node less than or equal to the upper limit B. B must be greater than or equal to the number of web objects divided by the number of nodes. The FIB size of each node after aggregation will depend on the location of originals of web objects on the network topology. However, we do not consider the location of each node on the network topology, and we leave the FIB aggregation with considering the node location as the future work.

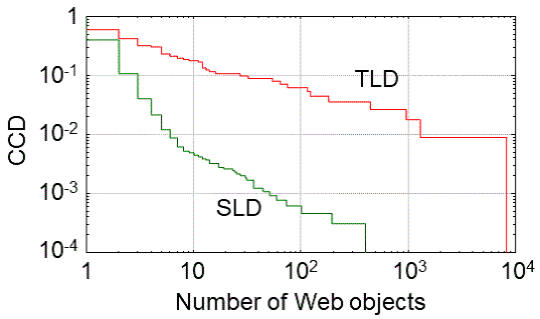


Fig. 3. Complementary cumulative distribution of web object count with identical TLD and identical SLD

TABLE IV
NUMBER OF URLS WITH EACH OF TOP 10 TLDS

Rank	TLD	N_d	Rank	TLD	N_d
1	com	8,241	6	io	122
2	net	1,292	7	uk	115
3	org	953	8	au	75
4	edu	441	9	co	64
5	gov	181	10	ca	54

TABLE V
NUMBER OF URLS WITH EACH OF TOP 10 SLDS

Rank	SLD	N_d	Rank	SLD	N_d
1	com/amgdgt	397	6	au/com	59
2	net/omtrdc	395	7	net/2o7	51
3	net/openx	193	8	com/gstatic	44
4	uk/co	100	9	com/netdna-ssl	36
5	com/mktoresp	73	10	net/fastly	36

B. Proposed Algorithm Placing Content at Nodes

Based on the policy described in Section V-A, we proposed an algorithm placing content at nodes so that the FIB size is further reduced. We assume that CDN cache servers are provided at all nodes of the network. We sort the TLDs in the descending order of the number of URLs using each TLD, and we define $D_1(x)$ as the x-th ranked TLD. We also sort the SLDs with each TLD in the descending order of the number of URLs using each SLD, and we define $D_2(s, y)$ as the y-th ranked SLD with s as the TLD. We define M_1 and $M_2(s)$ as the number of distinct TLDs and the number of distinct SLDs with s as the TLD.

Moreover, let $\boldsymbol{U}_1(x)$ denote the set of URLs with $D_1(x)$ as the TLD, and let $\boldsymbol{U}_2(s, x)$ denote the set of URLs with s as the TLD and $D_2(s, x)$ as the SLD. We define $m_1(x)$ and $m_2(s, x)$ as the number of URLs included in $\boldsymbol{U}_1(x)$ and $\boldsymbol{U}_2(s, x)$. Let A_n denote the number of URLs which can be placed at node n, and let n^* denote the node which has the maximum value of A_n[4]. Now, we describe the algorithm (Algorithm 2) placing content at nodes for reducing the FIB size.

Algorithm 2 Placing content at nodes for reducing FIB size

```
Initializes x = 1 and A_n = B for all n
while x ≤ M_1 do
    Finds n*
    if n_1(x) ≤ A_{n*} then
        Places U_1(x) at node n* and updates A_{n*} to A_{n*} − n_1(x)
    else
        Initializes y = 1
        while y ≤ M_2(D_1(x)) do
            while n_2(D_1(x), y) > A_{n*} do
                Places U_2(D_1(x), y) at node n* and updates A_{n*} to A_{n*} − n_2(D_1(x), y)
            end while
            Increments y and finds n*
        end while
    end if
    Increments x
end while
```

In the descending order of $m_1(x)$, we allocate $\boldsymbol{U}_1(x)$ to node n^* if $n_1(x) \leq A_{n^*}$. If $n_1(x)$ A_{n^*}, we divide $\boldsymbol{U}_1(x)$ into $\boldsymbol{U}_2(D_1(x), y)$, $y = 1, 2, \cdots, M_2(D_1(x))$, and we allocate $\boldsymbol{U}_2(D_1(x), y)$ to node n^* for each y. As a result, we can expect to improve the possibility of aggregating FIB entries at each node while satisfying $u_n \leq B$ where u_n is the number of URLs placed at node n.

C. Numerical Evaluation

We set B, the upper bound of object count placed at one node, to $W/N * 1.1 = 1,091$ where W is the number of web objects, i.e., 11,908, and N is the number of nodes, i.e., 12. Table VI shows the number of TLDs, SLDs, and URLs assigned to each node. As shown in Tab. IV, the TDL of about 70% URLs was "com", and these 8,241 URLs were assigned to nodes 1, 2, $\cdots$, 8 in the unit of SLDs. Moreover, 1,292 URLs with "net"as the SLD were assigned to nodes 9 and 10. At node 11, all URLs with "org"as the TLD were assigned. From these results, we confirmed that the proposed algorithm allocated URLs with the identical TLD or SLD to the same node with the constraint $u_n \leq B$.

Next, we applied Algorithm 1 described in Section IV-B to the placement of web objects obtained by Algorithm 2. Table VII summarizes $E^I_{d,n}$, the number of FIB entries without

[4]If there are multiple nodes having the maximum A_n, we randomly selects one of them as n^*.

aggregation, $E^{A,f}_{d,n}$, the number of FIB entries with aggregation in the forward direction, $E^{A,r}_{d,n}$, the number of FIB entries with aggregation in the reverse direction, $\Delta E^{A}_{d,n} \equiv E^{A,f}_{d,n} - E^{A,r}_{d,n}$, $R_{n,f}$, the reduction ratio of $E^{A,f}_{d,n}$, and $R_{n,r}$, the reduction ratio of $E^{A,r}_{d,n}$, based on the location of web objects designed by the proposed method. We define $R_{n,f}$ and $R_{n,r}$ as the reduction ratio of the FIB entry count obtained by designing the object location by the proposed algorithm compared with that obtained by the measured location of web objects, and they are given by $R_{n,f} = (E^{A,f}_{o,n} - E^{A,f}_{d,n})/E^{A,f}_{o,n}$ and $R_{n,r} = (E^{A,r}_{o,n} - E^{A,r}_{d,n})/E^{A,r}_{o,n}$.

$\Delta E^{A}_{d,n}$ was larger than $\Delta E^{A}_{o,n}$, and aggregating FIB entries in the forward direction was better than that in the revere direction when designing objects location by the proposed method. As shown in Tab. II, the number of web objects placed at node 5 was much larger than those at the other nodes in the measured location of web objects. whereas the number of objects allocated at node 5 was almost equal to those at the other nodes after placing the objects by the proposed method. Therefore, $E^{I}_{o,5}$, the initial number of FIB entries at node 5 in the measured allocation of objects, was much smaller than $E^{I}_{d,5}$, that in the proposed allocation, so the number of FIB entries at node 5 after the aggregation in the proposed allocation was larger than that in the measured allocation. However, at all the other nodes, the number of FIB entries was reduced by about 20% to about 90% by designing the object location by the proposed method. The average FIB size at each node after the entry aggregation was 3,394.0 in the measured location of web objects, whereas it was 1,848.4 in the location of web objects designed by the proposed method. Using the proposed design method of location of web objects, we can reduce the FIB size by about 45%.

TABLE VI
NUMBER OF TLDS, SLDS, AND URLS ASSIGNED TO EACH NODE

Node	#TLD	#SLD	#URL	Node	#TLD	#SLD	#URL
1	0	65	1,090	7	0	1,091	1,091
2	0	315	1,090	8	35	608	744
3	0	486	1,090	9	0	145	1,091
4	0	545	1,090	10	38	198	744
5	0	630	1,091	11	1	0	953
6	0	1,091	1,091	12	36	0	743

VI. CONCLUSION

In this paper, we firstly proposed an algorithm aggregating FIB entries in the unit of URL component in the forward and reverse direction. Through the numerical evaluation of the actual location of web objects, we confirmed that almost all the aggregation effect of FIB entries can be obtained by aggregating at the TLD and the SLD. Therefore, we also proposed an algorithm to allocate web objects to CDN cache servers so that the URLs with the identical TLD or SLD at the same node to effectively improve the aggregation effect of FIB entries. Through the numerical evaluation, we also showed that the FIB size can be reduced by about 45% by using the proposed allocation method of web objects.

TABLE VII
NUMBER OF FIB ENTRIES WITHOUT AGGREGATION, $E^{I}_{d,n}$, WITH AGGREGATION IN FORWARD DIRECTION, $E^{A,f}_{d,n}$, WITH AGGREGATION IN REVERSE DIRECTION, $E^{A,r}_{d,n}$, $\Delta E^{A}_{d,n} \equiv E^{A,f}_{d,n} - E^{A,r}_{d,n}$, REDUCTION RATIO OF $E^{A,f}_{d,n}$, $R_{n,f}$, AND REDUCTION RATIO OF $E^{A,r}_{d,n}$, $R_{n,r}$, BASED ON LOCATION OF WEB OBJECTS DESIGNED BY PROPOSED METHOD

Node	$E^{I}_{d,n}$	$E^{A,f}_{d,n}$	$E^{A,r}_{d,n}$	$\Delta E^{A}_{d,n}$	$R_{n,f}$	$R_{n,r}$
1	10,818	1,521	1,521	0	0.599	0.583
2	10,818	1,330	1,330	0	0.479	0.475
3	10,818	2,326	2,273	53	0.379	0.366
4	10,818	3,608	2,822	786	0.203	0.38
5	10,817	3,710	3,367	343	-0.219	-0.118
6	10,817	1,401	1,348	53	0.406	0.437
7	10,817	3,249	2,906	343	0.386	0.442
8	11,164	2,291	2,238	53	0.498	0.504
9	10,817	112	112	0	0.946	0.946
10	11,164	1,062	1,062	0	0.747	0.748
11	10,955	1,297	1,244	53	0.586	0.604
12	11,165	274	221	53	0.813	0.851

ACKNOWLEDGEMENTS

This work was partly supported by ROIS NII Open Collaborative Research 2019-FA02 and KDDI Foundation Research Grant Program 190051.

REFERENCES

[1] A. Afanasyev, C. Yi, L. Wang, B. Zhang, and L. Zhang, SNAMP: Secure Namespace Mapping to Scale NDN Forwarding, IEEE Global Internet Symposium 2015.
[2] Alexa webpage, https://www.alexa.com/siteinfo
[3] O. Ascigil, S. Rene, I. Psaras, and G. Pavlou, On-Demand Routing for Scalable Name-Based Forwarding, ACM ICN 2018.
[4] M. Butkiewicz, H. V. Madhyastha, and V. Sekar, Understanding Website Complexity: Measurements, Metrics, and Implications, ACM IMC 2011.
[5] R. Chiocchetti, D. Rossi, and G. Carofiglio, Exploit the Known or Explore the Unknown? Hamlet-Like Doubts in ICN, ACM ICN 2012.
[6] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi, A Survey on Content-Oriented Networking for Efficient Content Delivery, IEEE Commun. Mag., vol.49, no.3, pp.121-127, Mar. 2011.
[7] A. Detti, M. Pomposinim N. Blefari-Melazzi, and S. Salsano, Supporting the Web with an information centric network that routes by name, Elsevier Computer Networks, Vol. 56, No. 17, pp. 3705-3722, Nov. 2012.
[8] M. Gritter and D. R. Cheriton, An architecture for content routing support in the Internet, USENIX USITS 2001.
[9] Software is hard, http://www.softwareishard.com/blog/har-viewer/
[10] A. Hoque, et al., NLSR: Named-data Link State Routing Protocol, ACM ICN 2013.
[11] Internet 2, https://www.internet2.edu
[12] V. Jacobson, et al., Networking Named Content, ACM CoNEXT 2009.
[13] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, A data-oriented (and beyond) network architecture, ACM SIGCOMM 2007.
[14] MaxMind, GeoIP Downloadable Databases, https://dev.maxmind.com/geoip/legacy/downloadable/.
[15] E. Nygren, R. K. Sitaraman, and J. Sun, The Akamai Network: A Platform for High-Performance Internet Applications, ACM SIGOPS 2010.
[16] A. Rodrigues, P. Steenkiste, A. Aguiar, Analysis and Improvement of Name-based Packet Forwarding over Flat ID Network Architectures, ACM ICN 2018.
[17] K. V. Katsaros, W. K. Chai, and G. Pavlou, Bloom Filter based Inter-domain Name Resolution: A Feasibility Study, ACM ICN 2015.

[18] T. Schmidt, S. Wolke, N. Berg, and M. Wahlisch, Let ' s Collect Names: How PANINI Limits FIB Tables in Name Based Routing, IFIP Networking 2016.
[19] W. So, A. Narayana, and D. Oran, Named data networking on a router: Fast and DoS-resistant forwarding with hash tables, ACM/IEEE ANCS 2013.
[20] G. Xylomenos, et al., A Survey of Information-Centric Networking Research, IEEE Communications Survey and Tutorials, Vol. 16, No. 2, pp.1024-1049, 2014.
[21] L. Zhang, et al., Named Data Networking (NDN) Project, Technical Report NDN-0001, Oct. 2010.